UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Centre de la Imatge i la Tecnologia Multimèdia

Cinematica Quick Time Events

Bushidō

Trabajo Final de Grado

Grado en Diseño y Desarrollo de Videojuegos

Apellidos: Ortiz Peirano Nombre: Javier Ariel

Plan: Plan de estudio 2014

Director: Ripoll Tarré, Marc

Índice

Indice de tablas

Índice de figuras

Glosario

QTE : Quick Time Events, Cinematica interactiva.

Cinemática: Escena no jugable de un videojuego

Storyboard: Guión gráfico, sucesión de dibujos para entender una escena.

Body mechanics: Estudio del movimiento del cuerpo humano.

Blocking : Primera iteración de un proyecto , en animaciones la primera fase de animación.

Pipeline: Segmentación de un trabajo para optimizar su rendimiento.

Malla poligonal: Malla de polígonos que generan un objeto.

High poly / Low poly : Resolución de un modelo 3D.

Texturizado: Fase de aplicar una textura(color) a una malla Poligonal.

PBR: Estilo de texturizado realista.

Unwrapping: Proceso para extraer las coordenadas de texturas de un modelo.

Rigging: Proceso para crear los huesos de un modelo 3D.

Skinning: Proceso de unir la malla poligonal con los huesos del modelo 3D.

Hand painted: Estilo de texturizado de aspecto cartoon y poco realista.

Gameplay: Sección de una partida de un videojuego.

Metacritic: Página web de críticas online.

RPG: Role Play Game, género de videojuego de Rol.

Playtest: Probar un videojuego para confirmar su fiabilidad.

Storytelling: Arte de contar una historia.

Props: Modelos 3D que se utilizan para decorar el mapa, no se puede interactuar con ellos.

White Boxing: Proceso de diseño que se centra en crear un mapa a base de cajas blancas para probar el diseño.

UI: User Interface, Interfaz de usuario de un software con el cual interactúa el individuo.

Beta: Prototipo de un juego con todos sus aspectos jugables pero con algunos fallos.

Bug hunting: Buscar y arreglar Bugs de un videojuego.

Bug: Fallo dentro de un videojuego.

Assets: Modelos 3D con los cuales se puede interactuar.

Bushido: Camino del Guerrero Japonés.

Asano Naganori: Amo de los 47 Rōnin condenado a muerte.

Shōgun: Gobernante Japonés.

Daimyo: Soberano Feudal Japonés.

Kira Yoshinaka: Daimyo que entrenaba a Asano.

Seppuku/ Hara-Kiri: Ritual de suicidio Japonés.

Samurai: Guerrero espadachín Japonés.

Oishi Yoshio:Líder de los 47 Rōnin.

Edo: Nombre de la actual Ciudad de Tokio.

Sengakuji: Templo donde están enterrados los 47 Rōnin

LDD: Level Design Document.

Shuriken: Arma Arrojadiza con forma de estrella.

Katana: Espada tradicional Japonesa.

Sketch: Boceto rapido.

Joint: Hueso del esqueleto de animación

1. Introducción

El bushidō, o el camino del guerrero, fue un código ético muy estricto y peculiar seguido por los autodenominados "Bushi" (Samuráis) en el Japón feudal. En el país Nipón se cuenta una leyenda sobre cómo un grupo de samuráis dio la mayor muestra de lealtad a este código: La leyenda de los 47 Rōnin.

El objetivo de este trabajo es llevar la leyenda al ámbito de los videojuegos con una cinemática que nos explicará la historia de cómo tales guerreros demostraron a todo japón el verdadero sentido del camino del guerrero.

1.1 Motivación

Tom y Jerry, Bugs Bunny, Agallas el perro cobarde, Bob Esponja, son solo algunos ejemplos de los muchos dibujos animados que he consumido en mis primeros años, luego vino el boom de los videojuegos y solo tenía ojos para ellos. Los videojuegos no son solo un medio de entretenimiento, sino que también son un medio a través del cual poder expresarse, tal como hizo la pintura, la música o el cine. Estamos ante el canal artístico del siglo XXI. Y es aquí el porqué apasiona a multitud de personas alrededor del mundo, cada videojuego es una experiencia y cada persona puede sentirse identificada de manera distinta aun habiendo jugado al mismo videojuego. Es por eso que ahora con 22 años quiero dedicarme a desarrollar videojuegos, quiero ser Animador 3D.

De esta pasión por los videojuegos nace la motivación de utilizar este trabajo para experimentar todo el proceso de creación de una cinemática de un videojuego hasta que finalmente la ves, o interactúas con ella en el caso de este trabajo.

1.2 Formulación del problema

La leyenda de los 47 Rōnin es conocida por todo Japón y ha sido plasmada en diferentes medios:

- Relato adaptado del escritor argentino Jorge Luis Borges, que vio la luz en el primer volumen de cuentos publicado por Borges, *Historia universal de la infamia*, en el año 1935.
- Kenji Mizoguchi dirigió una película titulada *Los cuarenta y siete samuráis*, también conocida como *Los leales 47 Rōnin*, en el año 1941.

- Keanu Reeves protagonizó el papel principal de una adaptación cinematográfica del mismo nombre, *47 Rōnin*, dirigida por Carl Rinsch y estrenada en el año 2013.
- En 2015 se estrenó la película *Last Knights*, una adaptación dirigida por Kazuaki Kiriya que adapta la historia japonesa a una sociedad medieval ficticia que tiene como referente la cultura europea.

A pesar de tener unas cuantas adaptaciones en formato digital, nunca se ha llegado a hacer un videojuego basado en la leyenda nipona.

Por tanto, este proyecto pretende adaptar la historia al medio más popular del siglo XXI, los videojuegos.

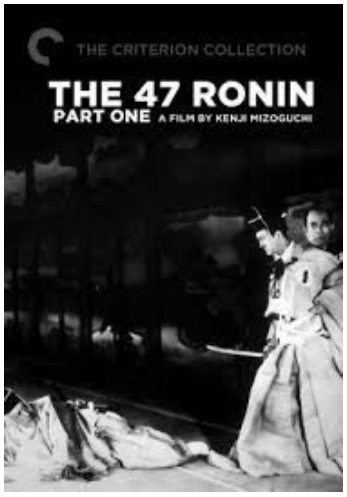


Fig. 1 Poster de la película los 47 Ronin, 1941

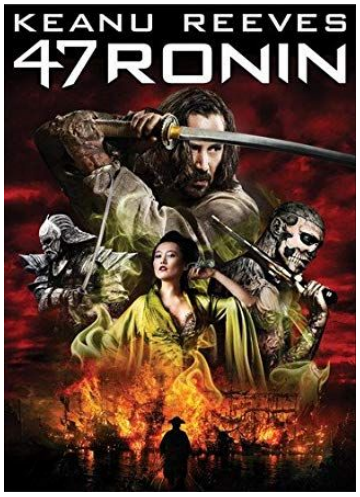


Fig. 2 Poster de la película 47 Ronin, 2013

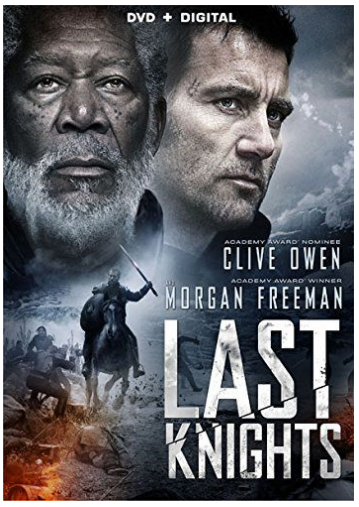


Fig. 3 Poster de la película Last Knights, 2015

1.3 Objetivos generales del TFG

El objetivo general de este trabajo es crear una cinemática interactiva mediante **QTE** que recree el asalto de los 47 Rōnin a la mansión de Kira. El jugador se pondrá en la piel de Ōishi Yoshio el líder de los Rōnin.

El trabajo pretende centrarse sobretodo en la Animación 3D, por lo que conseguir unas buenas animaciones que se complementan bien entre sí y tengan una buena fluidez y claridad es un objetivo primordial.

Hay que reafirmar que tanto la parte artística como la parte de diseño de niveles, no están consideradas como objetivos primordiales por lo que no se destinan grandes recursos ni se hará un trabajo especial en estos ámbitos, ya que como se ha dicho anteriormente en el trabajo se pretende crear una cinemática donde el punto fuerte sean las animaciones 3D.

1.4 Objetivos específicos del TFG

Los objetivos específicos para este trabajo son los siguientes:

- Realizar una buena planificación del proyecto.
- Realizar un Storyboard profesional (teniendo en cuenta mis capacidades artísticas) que incorpore todo lo que se verá en la cinemática final.
- Perfeccionar técnicas de modelado 3D.
- Realizar una estética uniforme en el proyecto.
- Perfeccionar conocimientos sobre animación tales como "body mechanics" o los 12 principios de animación.
- Realizar un correcto flujo de trabajo de animación: referencias, blocking, spline, polish.
- Conseguir un acabado final de buena calidad.

1.5 Alcance del proyecto

El alcance establecido de Bushidō es crear una cinemática interactiva de **QTE** que nos cuente cómo fue el asalto final a la casa de Kira.

El escenario consta de 3 zonas bien definidas, una introducción fuera de las puertas de la mansión, el asalto a la mansión y por último el desenlace final en el bosque. Se crearán 3 modelos diferentes para los samurais.

- El personaje principal
- Los compañeros y los enemigos (se utilizarán texturas diferentes para diferenciarlos)
- Kira, el enemigo final.

Al tratarse de un trabajo sobre una leyenda japonesa, va dirigido a personas que tienen un mínimo interés por la cultura asiática. Su jugabilidad se basa en **QTE**, así que su barrera de entrada es muy baja por lo que podrán jugarlo tanto jugadores habituales como jugadores esporádicos. A pesar de esto, está recomendado para jugadores adolescentes y adultos ya que contiene violencia explícita.

Dado que será una cinemática de duración corta y no un videojuego el cual se pueda comercializar, el mayor beneficiario seré yo mismo, consolidando todos los conocimientos que he ido adquiriendo durante los 4 años que ha durado mi formación académica, además de las personas que lo jueguen y puedan conocer esta leyenda japonesa.

2. Estado del Arte

Para poder ver en qué punto se encuentra la animación por ordenador a día de hoy, primero debemos entender qué es la animación y sus orígenes.

La Animación es el proceso que utilizan ciertas personas, en este caso animadores, para dar la sensación o crear la ilusión óptica de movimiento a imágenes, objetos inanimados, dibujos, etc.

La palabra Animación deriva del latín "animare" que significa dar vida a algo, por lo que muchos animadores consideran la animación como dar vida a personajes, con sus respectivas personalidades, carácter, etc.

Las primeras animaciones tal como las conocemos ahora datan del principio del siglo XX, pero no fue hasta la era de oro de la animación en Estados Unidos (desde 1928 hasta 1972), donde gracias a grandes animadores y sus dibujos animados fomentaron la industria y a consolidar una base sólida para todos los animadores que llegaron después.

En 1981, los animadores de Disney Studios Ollie Johnston y Frank Thomas, publicaron un libro titulado Disney Animation: The Illusion of Life en el cual se detalla el proceso de animación que se utilizaba en la compañía durante la era de oro de la animación.

En este libro se recogen los 12 principios de animación.

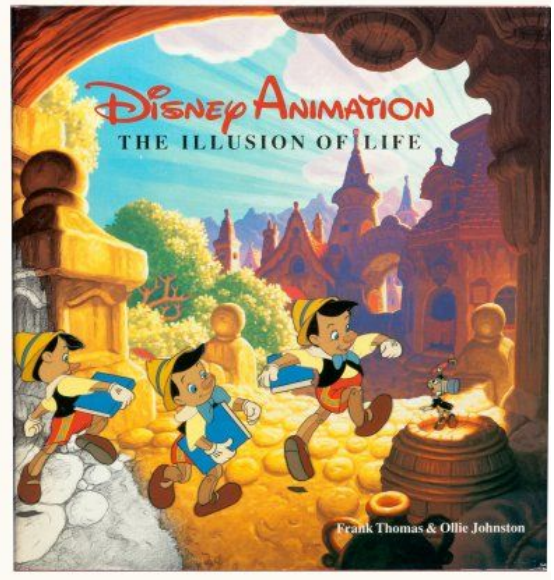


Fig. 4 Portada Disney Animation The illusion of life, 1981

Los 12 Principios de animación

Squash and Stretch

El propósito de Estirar y Encoger es el de dar sensación de peso y flexibilidad a los objetos además de exagerar el movimiento en la dirección deseada.

Anticipation

La anticipación se utiliza para preparar al espectador para la siguiente acción a realizar, por ejemplo agacharse antes de un salto.

Staging

La puesta en escena es similar a la utilizada en el cine o el teatro, y pretende dirigir la atención del espectador a lo que es relevante dentro de la toma.

Straight Ahead Action and Pose to Pose

La Animación Directa y la Animación Pose a Pose, son dos frameworks diferentes a la hora de animar. La Animación directa es aquella que empieza en el proceso de animación desde el primer frame y se dibujan los siguientes directamente, mientras que la Animación Pose a Pose, se dibujan unos fotogramas claves separados entre sí y posteriormente se rellenan los espacios.

Follow Through and Overlapping Action

La Acción complementaria y Acción superpuesta están muy fuertemente relacionadas, ya que gracias a ambas conseguimos más realismo en las animaciones y de que todo se rija por las leyes de la física.

La Acción complementaria describe qué es lo que harán los objetos después de un movimiento, es lo complementario a la anticipación.

Mientras que la Acción superpuesta cubre la noción de las diferentes partes de un cuerpo que se mueven a diferentes rangos, por ejemplo en un puñetazo la cabeza y el brazo se mueven a distintas velocidades, pero ambos acompañan el movimiento central.

Slow In and Slow Out

La aceleración y la desaceleración son conceptos que tenemos muy presente en nuestro día a día. Un coche no se pone a 100 km/h instantáneamente sino que necesita acelerar, lo mismo pasa con el movimiento de las animaciones, ya que estas necesitan un tiempo tanto para iniciar la acción como para finalizarla.

Arc

Los movimientos tienden a seguir una trayectoria de Arco, como los brazos balanceándose al caminar. Los movimientos que no siguen un arco suelen apreciarse como no naturales.

Secondary Action

Las acciones secundarias se utilizan para complementar y enfatizar la acción principal del personaje, añadiendo más detalle y haciéndola más vistosa.

Timing

El Timing es esencial en una animación. Es el número de dibujos que se utilizan para una acción, es por así decirlo la velocidad, cuanto menos dibujos haya más rápido se moverá un objeto, y si hay muchos dibujos por medio, más lento se moverá. Es vital para darle realismo y peso a los objetos.

Exaggeration

Si las animaciones plasmasen la realidad tal y como es, estas se verían aburridas. En la animación buscamos crear mejoras de la realidad, para eso se utiliza la exageración.

Solid drawing

El dibujo sólido consiste en tener en cuenta que se está intentando representar un espacio 3D en una pantalla 2D, por lo que hay que respetar las proporciones, además de darle volumen y peso.

Appeal

El atractivo de un personaje es vital para su carisma. El dibujo debe darle al espectador la sensación de que el personaje es interesante y tiene una buena historia que contar. Hay muchas técnicas referidas a las formas que se utilizan para dibujar, pero no entraremos en detalle.

A pesar de que estos principios se establecieron hace casi 40 años, siguen vigentes hoy en día, y se consideran la biblia de la animación.

Pero la sociedad avanza y la tecnología con esta, la llegada de los ordenadores revolucionó la vida de muchas personas y también dio pie a nuevas tecnologías que los animadores aprovechan para llevar su trabajo un paso más allá.

La Animación por Ordenador

La Animación por Ordenador se empezó a utilizar comúnmente a finales del siglo XX, pero no logró evolucionar significativamente hasta que en 1995 el estudio de animación Pixar estrenó la película Toy Story, el primer largometraje enteramente de Animación por Ordenador.

A partir de ese momento, hemos tenido multitud de largometrajes de Animación 3D, y no es raro ver cada fin de semana un nuevo estreno.

En los videojuegos fue un poco más complicado,ya que no hubo un juego que fuera totalmente innovador por incorporar Animación 3D, sino que fueron muchos juegos que poco a poco comenzaron a experimentar con los gráficos en 3D, gracias a esto la industria avanzó hasta el punto en que nos encontramos hoy en día.

La Animación 3D a Día de Hoy

La creación de una cinemática pasa por muchos procesos hasta que finalmente sale a la luz. En la industria se sigue una pipeline de Preproducción, Producción y finalmente Postproducción (lo veremos en detalle en metodología). Así que empezaremos por lo esencial y en especificar los apartados en los que se divide una cinemática, los softwares que se utilizan y su correcta utilización

Modelado 3D

Para conseguir una animación 3D, lo primero que necesitamos es un personaje al cual poder animar, este proceso de creación recibe el nombre de Modelado 3D.

El modelado 3D consiste en crear y dar forma a mallas de polígonos en un espacio 3D. Gracias a esto podemos tener un personaje fácilmente reconocible que al fin y al cabo no es nada más que una serie de vértices conectados entre sí.

Actualmente hay 2 tipos de modelado que podemos realizar un objeto o personaje.

Modelado Poligonal

El modelado Poligonal consiste en utilizar los vértices moviéndose en un espacio 3D para formar una malla poligonal que representa un objeto. Cuanto mayor sea el número de polígonos mayor definición tendrá un modelo. Actualmente la mayoría de videojuegos utilizan mallas con un alto número de polígonos, denominados High-Poly. Pero, también hay juegos que experimentan con las formas y con modelos con un menor número de polígonos, denominados Low-Poly donde igualmente se consiguen resultados muy buenos.

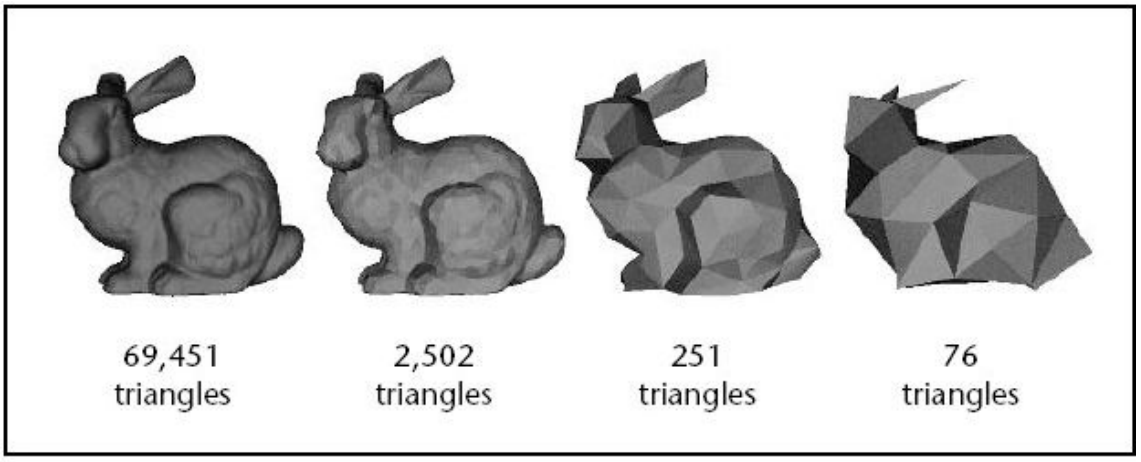


Fig. 5 Niveles de detalle modelado 3D

Escultura Digital

La escultura digital es parecida al modelado poligonal, pero aquí no utilizamos los vértices sino que es más parecida a la escultura tradicional, donde gracias unos "pinceles" agregamos o quitamos superficie como si de una obra de arcilla se tratase.

Este tipo de modelado es muy popular gracias a su hiperrealismo y la libertad artística que conlleva, comúnmente utilizado en los videojuegos para personajes orgánicos.

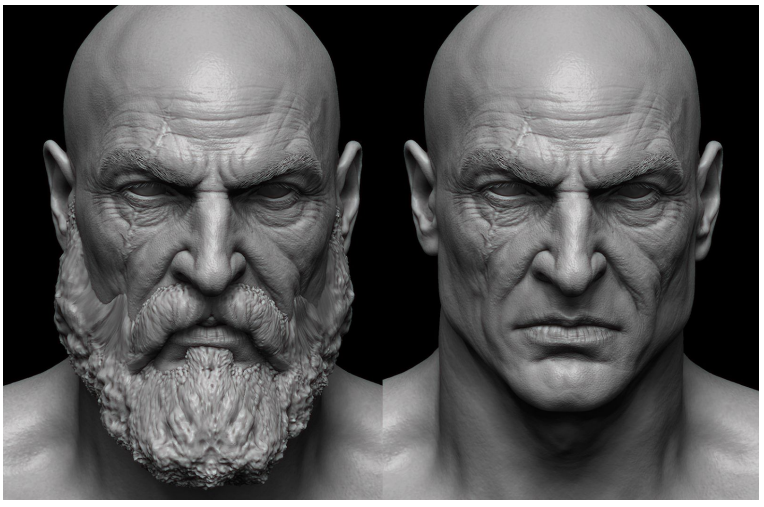


Fig. 6 Escultura Digital Kratos, Raf Grassetti

Texturizado

Una vez hemos modelado al personaje, este será de color gris, ya que no hemos dado ningún tipo de color a esta malla, aquí es donde entra el Texturizado, una parte muy importante del arte digital que a veces no se le presta mucha atención.

El texturizado consiste en darle una textura a nuestra maya, ya sean colores planos o materiales que más tarde reflejaran la luz o representaran hendiduras.

Aunque actualmente cada videojuego utiliza su estilo artístico hay dos tipos de texturizado que predominan.

Hand-Painted

El Hand-Painted o pintado a mano es la técnica de texturizado que solo utiliza un mapa de textura el Diffuse Map o mapa de color, esta técnica se utiliza normalmente en videojuegos de estilo cartoon o poco realistas por ejemplo League of Legends. Normalmente los juegos más realistas utilizan la técnica PBR.

Fig. 7 Modelo Hand Painted Zed, League of Legends

PBR

El PBR(*Physically based rendering*) es una técnica de texturizado la cual a diferencia del Hand-Painted, esta busca recrear con algoritmos matemáticos el comportamiento de la luz en el mundo real. Se utiliza para dar un resultado mucho más realista al utilizar utiliza diversos mapas de texturas:

Diffuse: El mapa de textura básico, es el que le da color a nuestro objeto. Es tener una foto del material que queremos que represente nuestra textura.

Albedo: Similar al Diffuse pero en este no contamos con sombreado, se utiliza para darle una mayor definición a la textura, aunque a veces se suele poner en el sitio del Diffuse.

Ambient Occlusion: El mapa contrario al albedo, en este contamos con toda la información de las sombras que proyecta nuestra textura.

Normal: El mapa de normales proporciona la dirección de cada normal de los polígonos de nuestra malla, es decir la dirección hacia donde está orientado cada polígono.

Displacement: El mapa de desplazamiento nos permite deformar nuestra malla para darle volumen, si los volúmenes son demasiado grandes es mejor deformar la geometría directamente.

Gloss: El mapa de brillo nos proporciona la información de cómo se comportan los reflejos de nuestra malla.

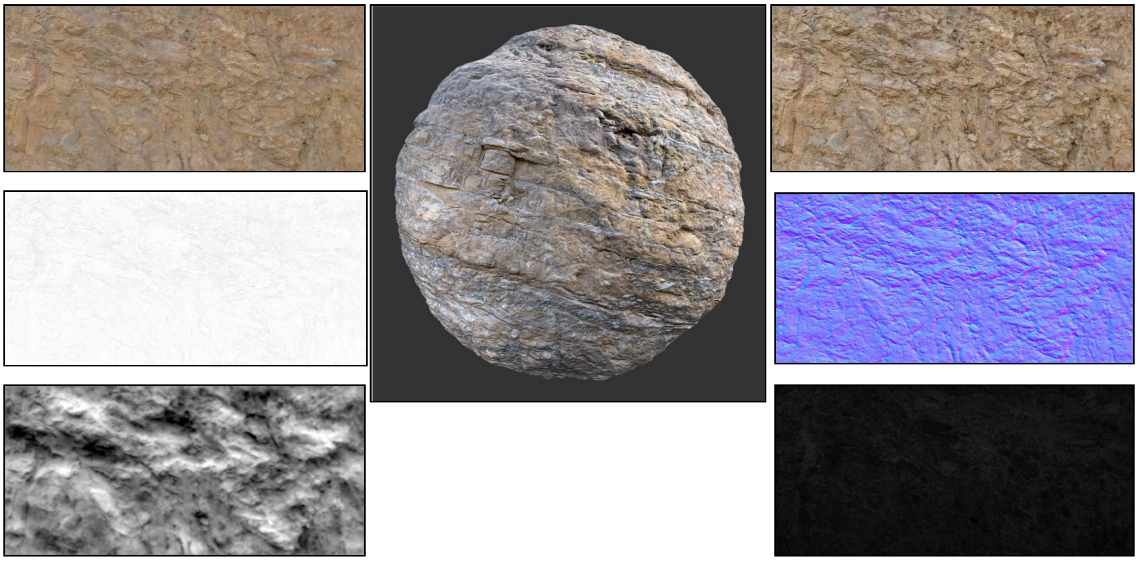


Fig. 8 Textura de Roca con sus respectivos mapas de izquierda a derecha

Ahora bien, qué hacemos para que una textura 2D se aplique correctamente a un modelo 3D, aquí es donde entran en juego las UV o coordenadas de textura.

UV Mapping

El mapeado de texturas consiste en el proceso por el cual una imagen 2D, en este caso la textura, se aplica a una malla poligonal 3D.

El proceso que se sigue es el de recortar por partes la malla y proyectarla en un plano 2D denominado **unwrapping**. Gracias a un algoritmo matemático, cada pixel de la textura es correctamente pintado en los pixeles del polígono 3D. Es un proceso similar al de envolver con papel de regalo.

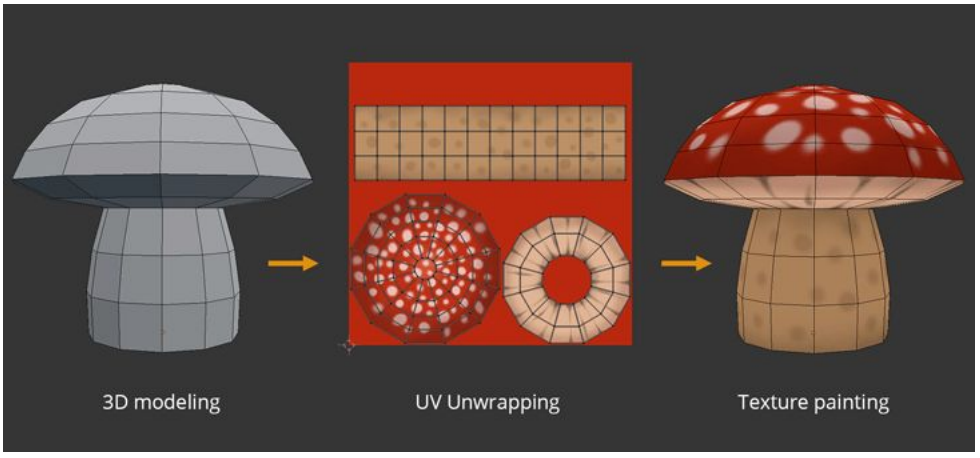


Fig. 9 Proceso de Texturizado y Unwrapping de una seta

Rigging

Ahora que tenemos un modelo 3D finalizado, debemos prepararlo para que se pueda animar. Pero, cómo conseguimos que un modelo 3D estático pueda moverse y realizar animaciones como las que vemos en los videojuegos.

Aquí, es donde entra el proceso de Rigging o también conocido como esqueleto. No son más que un grupo de "huesos" que le indican a la malla 3D de qué manera deberá comportarse. Funciona de la misma manera que nuestro cuerpo, en este caso los músculos que van unidos al hueso es la malla 3D.

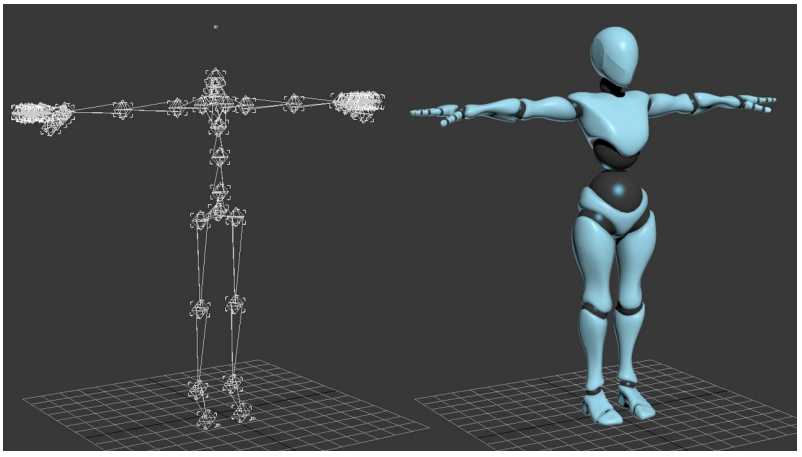


Fig. 10 Esqueleto de un Personaje

Skinning

El skinning es lo que complementa al Rigging, consiste en aplicar una capa de influencias sobre la malla 3D, es decir, decirle en qué medida seguirá a que hueso, es un proceso largo ya que un mal Skinning dará lugar a errores visuales que son fácilmente apreciables.

Fig. 11 Influencias del Esqueleto en la malla poligonal

Controladores

Para poder animar correctamente un personaje, se necesita una serie de controladores que puedan hacer la vida más fácil al animador, permitiéndole mover cada hueso del modelo sin necesidad de tener visión de él, para eso se utilizan los controladores.

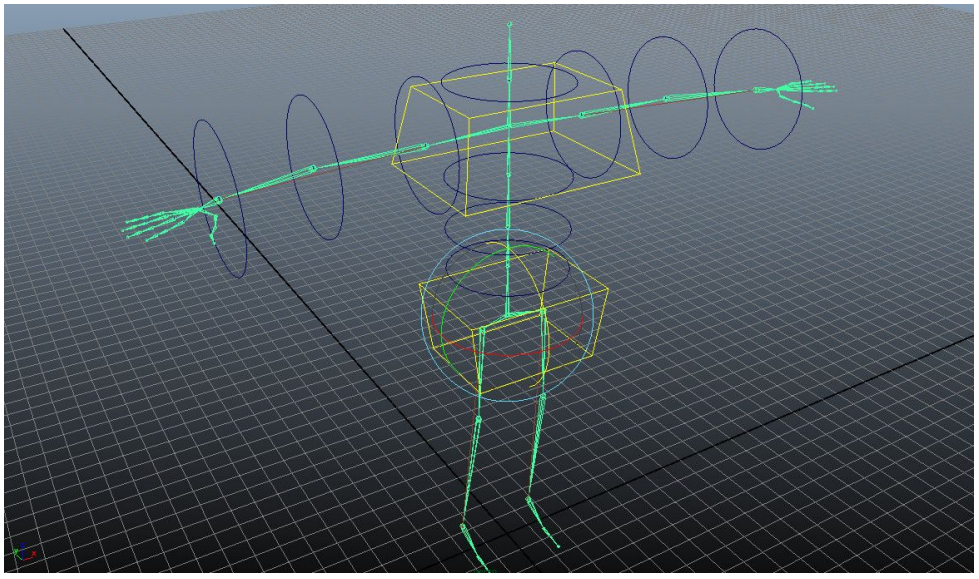


Fig. 12 Controladores sobre un Esqueleto

Animacion 3D

Una vez se ha pasado por todo el proceso de preparación, se pasa al proceso de animación, el cual se divide en 4 partes:

Referencias

La primera parte para realizar una animación 3D es pensar cómo va a ser esta animación, para ello se recogen o simplemente se realizan referencias de personas reales para ver cómo se comporta el cuerpo humano y para tener una especia de guia a la hora de animar.

Blocking

Una vez está definida y documentada la animación, se procede a hacer el blocking, es donde se coloca el personaje en las poses claves para poder controlar el Timing y Spacing de la animación. Al finalizar esta fase, se obtiene una animación robótica que transiciona entre poses de manera muy bruta, sin interpolación.

Spline

La tercera fase de una animación es en la cual procedemos a interpolar las poses para ver la animación fluida. En esta fase se mejora la animación, utilizando los 12 principios, se exagera, se utilizan aceleraciones etc, al final tenemos una animación casi terminada, pero queda la última fase.

Polish

La última fase y probablemente la más difícil es la de polish, donde se pule la animación hasta que queda a nuestro agrado, se procede a hacer las acciones secundarias, overlapping, etc.

Así es como se ha realizado la animación por ordenador desde que se inventó, ahora bien, gracias a los avances tecnológicos se ha conseguido desarrollar una técnica que evita las 3 primeras fases y solo hace falta pulir las animaciones.

Motion Capture

La captura de movimiento es una técnica de grabación de movimiento que se ha desarrollado a principios del siglo XXI. Esta técnica consiste en realizar un seguimiento de marcas situadas en un traje preparado en una zona condicionada. Es decir, se sigue cada punto del traje de un actor durante una grabación para así poder reproducir los movimientos que éste hace en un esqueleto 3D.

Esta técnica a pesar de ser muy cara es muy fiable ya que con ella podemos obtener resultados muy realistas al tratarse de plasmar directamente los movimientos de un ser humano en el ordenador.

Esta técnica se ha desarrollado más allá y también permite realizar captura de movimiento en expresiones faciales, así los animadores pueden obtener directamente los movimientos de los músculos de la cara de los actores al recitar el guión.

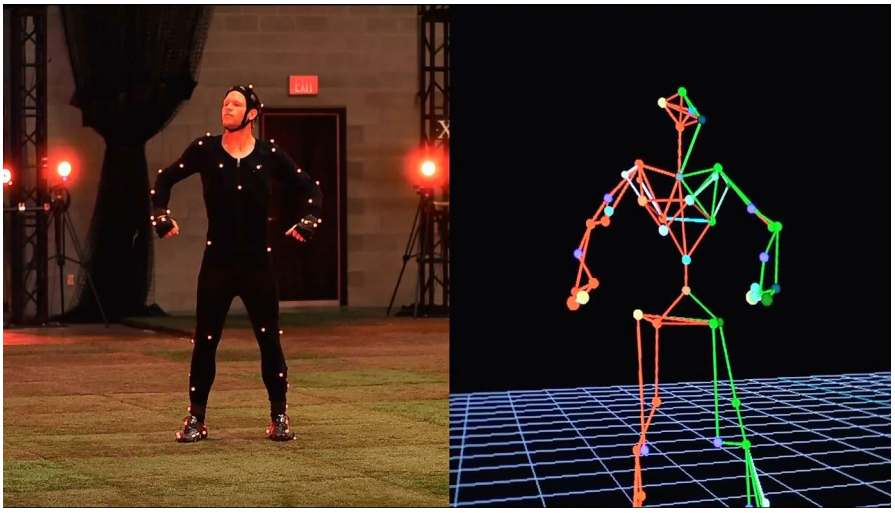


Fig. 13 Traje de Motion Capture y su Esqueleto digital

Fig. 14 Motion Capture Facial

Quick Time Events

Las cinemáticas interactivas (**QTE**) se basan en un simple cambio de escenas o de animaciones cuando el jugador interactúa con ésta, ya sea pulsando botones o moviendo el mando. Para realizarlas se utilizan máquinas de estados que al activarse intercambia o bien entre diferentes escenas o directamente entre las animaciones que posee un personaje.

Fig. 15 QTE, Ryse: Son of Rome

Software 3D

Después de ver todo el proceso que se utiliza profesionalmente para realizar una animación, hay que conocer qué programas se utilizan para realizar cada apartado.

Los programas actuales que utilizan los estudios para modelar son 3.

Autodesk 3ds Max, Autodesk Maya, Zbrush

3ds Max y **Maya** pertenecen a Autodesk y son programas de pago con suscripción, ambos se utilizan para el modelado poligonal, aunque tengan pequeñas diferencias, se pueden conseguir los mismos resultados con ambos.

En cambio **Zbrush** es de la empresa Pixologic, también es un programa de pago con suscripción, pero a diferencia de los programas de Autodesk, **Zbrush** se utiliza para escultura digital.

Existen alternativas para realizar modelado 3D, pero estos son los más populares entre los estudios de videojuegos.

Para el texturizado se acostumbra a utilizar **Substance Painter**, un programa de pago de Allegorithmic, aunque también se utiliza Zbrush o Photoshop.

Para **Skinning**, **Rigging**, **controladores** y **animación** se utiliza tanto **3ds Max** como **Maya**, esto depende del gusto del artista y con que programa se sienta más cómodo.

Ahora bien para realizar un videojuego se utiliza un motor de videojuegos, los más utilizados a día de hoy son **Unity** y **Unreal Engine**. Ambos son gratuitos y ofrecen características muy parecidas, la mayor diferencia es a cuestión de optimización y manera de trabajar.

Fig. 16 Logos(de izq. a der.) 3Ds Max; Zbrush; Maya; Substance Painter; Unity

Tecnología utilizada en Bushidō

Al tratarse de un trabajo de fin de grado, y teniendo en cuenta el tiempo y la limitación de personas trabajando en él, ya que solo lo realiza una, se han tomado decisiones de diseño acordes a estas variables.

El **modelado** y **texturizado** de Bushido será estilo low poly, y con una paleta de colores planos, no se realizarán modelado de rostros, así se quitará carga de trabajo en las animaciones. Pese a esto se puede conseguir un buen resultado combinando estos dos factores tal como hizo el videojuego *Ashen*. La diferencia es que *Ashen* utiliza un handpainted falseado ya que son texturas planas pero combinadas con **PBR**.

En lo relacionado a las Animaciones se seguirá el modelo tradicional de workflow, es el modelo más estandarizado y ya lo he utilizado para anteriores proyectos. No se harán animaciones faciales ya que sería aumentar muchísimo la carga de trabajo por eso se ha decidido utilizar un estilo similar a *Ashen*.

En cuanto a la combinación entre **QTE**, se utilizaran a la vez cambios de **Animaciones** como cambios de escena, dependiendo del requisito de storytelling.

Tanto para el modelo como el **unwrapping** de **texturas** se utilizará **3Ds Max**, ya que es el programa con el que he trabajado durante la carrera y con el que me siento más cómodo.

Para el **texturizado** se empleará **Substance Painter** y **Zbrush**, dependiendo de la necesidad del modelo se utilizará un software u otro.

En cuanto a **Rigging**, **Skinning**, **Controladores** y **animaciones**, se usará **Maya**, es el programa por excelencia de animación y es el más cómodo para trabajar además de tener experiencia previa con el.

Para toda la fase de programacion y creacion del mundo se utilizará el motor **Unity**, es el motor más flexible y con el que más cómodo me siento, he realizado múltiples proyectos en este motor y se adapta perfectamente a las necesidades del proyecto.

Fig. 17 Ashen,2018

2.1 Estudio de Mercado

Bushidō tiene tres pilares importantes, los **QTE** , ser un juego **low-poly** y su temática japonesa.

Dado que no encontramos ningún producto que junte estos tres pilares, hemos analizado a sus competidores por separado.

Quick Time Events

Hay muchos juegos que incluyen los QTE como un bonus para sus cinemáticas, sin embargo se quedan en eso, un simple bonus para su juego, mientras que otros como *Detroit: Become Human*, el cual basa todo su gameplay en pequeños movimientos del mando, pulsación de botones o movimientos en el touchpad.

Ha sido todo un éxito de ventas vendiendo un total de 2 millones de copias y obteniendo una puntuación de 7.8 en Metacritic.

Fig. 18 QTE, Detroit: Become Human, 2018

Japon

La temática japonesa es muy utilizada en todo el mundo, tanto en cine, libros, y videojuegos.

El 22 de marzo saldrá a la venta el último juego de *From Software* llamado *Sekiro: Shadows Die Twice*, un videojuego **RPG** de temática japonesa. *From Software* es muy famosa gracia a sus saga de videojuegos *Dark Souls* y está asentada como una de los referentes de juegos **RPG**.

Fig. 19 Sekiro Shadows Die Twice, 2019

Low-Poly

Estilo artístico **Low-Poly** lo suelen utilizar estudios indies que necesitan optimizar su trabajo, ya que al no disponer de un gran número de desarrolladores no pueden conseguir unos resultados tan buenos con el PBR.

Hay muchos juegos en el mercado con estilo **Low-Poly** pero el que más se asemeja el buscado en este trabajo sería *Ashen* y *Astroneer*. Ambos juegos acumulan buenas críticas y ventas.

Fig. 20 Astroneer, 2019

3. Gestión del proyecto

3.1 Procedimiento y herramientas para el seguimiento del proyecto

Este proyecto tiene una duración aproximada de 4 meses. Se estima que tendrá una carga de trabajo de 300 horas que se reparten entre los 5 meses que dura el trabajo. Como se explica más adelante, el proyecto se divide en 3 fases: **Preproducción**, **Producción** y **Postproducción**. Estas se reparte a lo largo de las entregas correspondientes del trabajo.

Fig. 21 Planificacion del Proyecto

3.1.1 GANTT

Para un correcto funcionamiento se utilizará un diagrama de *GANTT* el cual se elabora a partir de la herramienta de *Trello*

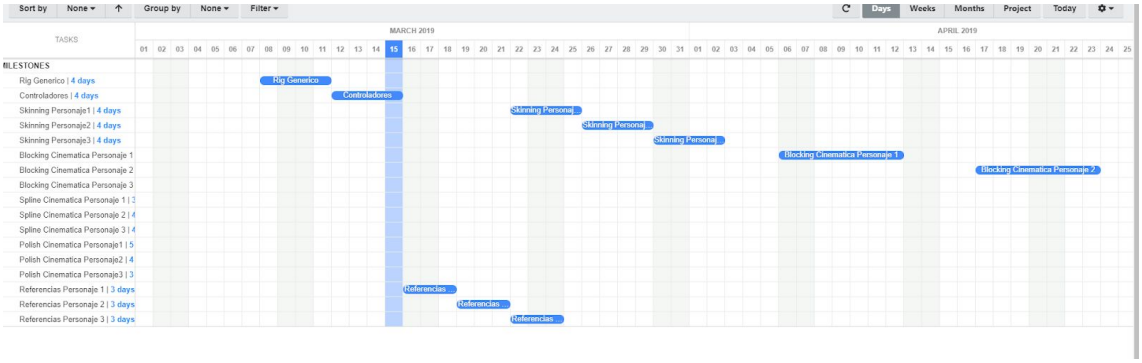


Fig. 22 Diagrama de Gantt Apartado Animacion

3.1.2 Trello

Para el seguimiento de Tareas se utilizará el software *trello*

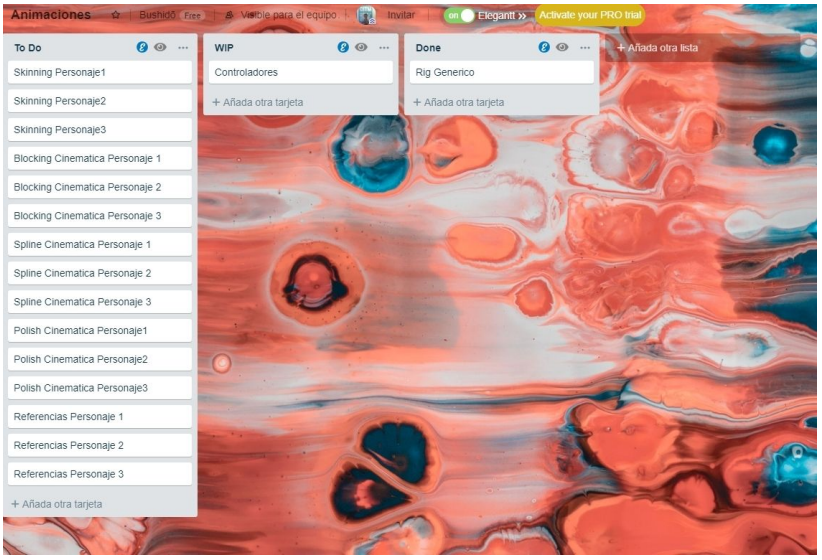


Fig. 23 Imagen de Trello Apartado Animacion

3.1.3 Google Drive

Para el almacenamiento de los archivos se utilizará el software **Google Drive**, una herramienta de *Google* que guarda archivos online para así acceder desde cualquier dispositivo.

3.2 Herramientas de validación

Para validar el correcto funcionamiento del proyecto, se realizará playtest tanto internos como externos.

Estos se basarán en dejar jugar libremente al jugador.

Solo se le hablará si él pregunta primero, así podemos ver si la cinemática es lo suficientemente intuitiva como para ser superada sin ningún inconveniente.

Una vez haya acabado de jugar, se le pedirá que rellene un formulario acorde a la zona del juego que haya jugado.

Esto se hará a través de Google Forms, una herramienta desarrollada por google la cual nos permite realizar encuestas y que todas queden almacenadas en una base de datos.

Acorde a los resultados de estas encuestas y a los comentarios recibidos en los playtest se podrá validar si la cinemática funciona bien o qué elementos deben cambiarse para tener finalmente un trabajo de calidad.

Además de esto se pedirá a profesores con conocimiento en el ámbito del CITM que den sus comentarios conforme vaya avanzando el proyecto, una visión de alguien dentro de la industria mejorará el proyecto notablemente.

3.3. DAFO

Todo proyecto tiene sus puntos fuertes y débiles, y hay que saber identificarlos y trabajar acorde a ellos, los de este proyecto son los siguientes:

Fortalezas	Debilidades
Pasión por la Animación Experiencia en Animación de Personajes Conocimientos de Software necesarios	Trabajo individual multidisciplinar Calidad no profesional Poca habilidad de programación y arte Poco conocimiento sobre Storytelling
Oportunidades	**Amenazas**
Popularidad tematica japonesa Portfolio Workflow Profesional	Producciones profesionales

Tabla 1. DAFO

Fortalezas

- **Pasión por la Animación**: La animación es el ámbito al cual me quiero dedicar desde el principio de la carrera, por lo cual va a ser muy gratificante realizar este trabajo.
- **Experiencia en Animación de Personajes:** He realizado trabajos de animación de personajes previamente, y conozco la metodología utilizada para el trabajo.
- **Conocimientos de Software necesario:** Domino los softwares que se utilizaran durante el desarrollo.

Oportunidades

- **Popularidad temática japonesa:** La temática japonesa es muy famosa y tiene muchos seguidores alrededor del globo.
- **Portfolio:** Este trabajo servirá para ampliar mi portfolio como animador.
- **Workflow Profesional** Este proyecto da la oportunidad experimentar con un workflow profesional y entender los problemas que cada apartado puede comprender.

Debilidades

- **Trabajo individual multidisciplinar:** Al ser solo una persona, todas las tareas recaen sobre mí.
- **Calidad no profesional:** Con el límite de tiempo y personal, es difícil llegar a una calidad profesional.
- **Poca habilidad de programación y arte:** No son una prioridad en el proyecto por lo cual se intentará solventar de la manera más fácil posible.
- **Poco conocimiento sobre Storytelling:** No poseo los conocimientos suficientes como para hacer un trabajo de Storytelling profesional.

Amenazas

- **Producciones profesionales:** Un estudio profesional podría realizar el mismo trabajo y con una calidad mayor en un tiempo mucho más bajo.

3.4. Riesgos y plan de contingencia

Es de vital importància detectar els riscos que poden posar en perill la feina i buscar solucions per en cas de ser necessari poder reconduir el projecte.

Riesgo	Solucion
Fallo del software y pérdida de datos	realizar copias de seguridad cada dia y guardado automático de los programas
No disponer de musica y efectos de sonidos originales	buscar efectos de sonido y música de libre disposición
No realizar correctamente Props	El escenario y diseño no son una prioridad, se utilizaria White Boxing
Pérdida de todo el material por problemas de hardware	Todo el material estará subido en la nube, para poder recuperarlo en cualquier momento.

Tabla 2. Riesgos y Soluciones

A todo esto se le añade el factor de no poder completar el proyecto total, como hemos comentado anteriormente el objetivo principal es el de realizar una cinemática de **QTE**, por lo que se priorizará la animación final de combate en caso de no poder completar la totalidad del proyecto, ya que es la más completa y donde se puede conseguir un resultado cerrado óptimo.

3.5. Análisis inicial de costes

Si enfocamos este proyecto como uno profesional, deberíamos calcular los costes de producción que este supone, para así tener una visión general de si este proyecto es viable o no, y en qué invertimos nuestro dinero.

Lo primero, sería pagar las licencias de cada programa que fuéramos a utilizar, en el apartado de Estado del Arte hemos comentados cuales utilizaremos así que hemos calculado su precio pagando mensualmente la suscripción.

Hemos hecho un caso hipotético donde dispondremos de 1 empleado por departamento, el cual cobra lo mismo que un becario de la UPC.

Por último, necesitamos una oficina en la cual poder realizar el trabajo, hemos calculado unos 600€ contando gastos como alquiler, internet, luz y agua.

Todo esto en un periodo de 5 meses, que es lo que dura el proyecto.

Presupuesto Inicial			
Licencias			
	Precio	Tiempo	Total
Maya	€248.05	5 Meses	€1,240.25
3DS Max	€248.05	5 Meses	€1,240.25
Zbrush	€39.95	5 Meses	€199.75
Substance P.	€19.90	5 Meses	€99.50
Unity	€125.00	5 Meses	€625.00
Personal			
1 Diseñador	6.5€/hora	600 Horas	€3,900.00
1 Modelador	6.5€/hora	600 Horas	€3,900.00
1 Animador	6.5€/hora	600 Horas	€3,900.00
1 Programador	6.5€/hora	600 Horas	€3,900.00
Oficina			
1 oficina	600€	5 meses	€4,000.00
Total		€23,004.75	

Tabla 3. Presupuesto Inicial

Un presupuesto de **€23,004.75** es lo mínimo que necesitamos si quisiéramos realizar nuestro proyecto, obviamente es sólo una suposición ya que las horas pueden variar y el coste de cada persona también.

4. Metodologia

Al tratarse de una obra audiovisual, utilizaremos la metodología por excelencia en este ámbito:

Preproducción, **producción** y **postproducción**.

Preproduccion

- Se realizará una investigación sobre la leyenda de los 47 ronin, adaptando su historia para que sea asequible de realizar en el TFG.
- Se realizará un storyboard con todas las escenas de la cinemática, haciendo hincapié en los puntos donde el jugador deba tomar las decisiones.
- Se realizará un estudio del estilo artístico óptimo para realizar el trabajo.
- Se realizará un listado de las animaciones que se deben recrear además de su duración.
- Se realizará un concepto del mapa jugable para así poder realizar el **white boxing** más adelante.

En esta fase se llevará a cabo todo el estudio pertinente para que no haya dudas a la hora de pasar a la fase de producción.

Produccion

Aquí es donde se concentra la mayor parte del trabajo, donde a partir de toda la información que tenemos de la preproducción procederemos a crear contenido

- Blocking del escenario donde se llevará a cabo la cinemática.
- Búsqueda de referencias tanto de modelado como de animación
- Modelado y texturizado de props y personajes
- Rigging y Skinning de los personajes.
- Animacion de la cinemàtica
- Máquina de estados de los QTE.

Al finalizar esta fase tendremos una **Beta** de nuestra cinemática, es decir todo está puesto en sus sitio, todo funciona pero quizás falte mejorar animaciones, arreglar bugs, o alguna textura.

Postproduccion

Esta es la última parte del proyecto, hasta aquí habríamos producido todo lo que finalmente saldrá en la cinemática, solo realizaríamos las siguientes acciones:

- Pulido de animaciones
- Elección de efectos de sonido y música ambiental
- Creación de UI para interactuar con los QTE.
- Creación de Cámaras.
- **Bug Hunting**

5.Desarrollo del proyecto

Cambios de Desarrollo

Antes de Comenzar el desarrollo se tiene que comentar los cambios que se han hecho antes de este.

Teniendo en cuenta la carga de trabajo y el objetivo del proyecto, se ha decidido comprar **Assets** en la **Asset Store** de **Unity**.

Se ha comprado "POLYGON - Samurai Pack" el cual tiene un precio de 25€ que se sumará a los 23,004.75€ quedando un total de 23,029,75€ de presupuesto.

Fig. 24 Imagen de Samurai Pack

Además de esto se ha aplazado el lanzamiento final de este proyecto hasta el sábado 31 de Agosto, por lo cual habrá que modificar el presupuesto final del proyecto además de las Herramientas de seguimiento del proyecto.

Presupuesto final

Como se ha comentado antes, el proyecto se aplaza hasta el 31 de Agosto, por lo que habrá que añadir dos meses mas de desarrollo al presupuesto inicial para obtener el Presupuesto Final del proyecto.

Presupuesto Final			
Licencias			
	Precio	Tiempo	Total
Maya	€248.05	7 Meses	€1,736.35
3DS Max	€248.05	7 Meses	€1,736.35
Zbrush	€39.95	7 Meses	€279.65
Substance P.	€19.90	7 Meses	€139.30
Unity	€125.00	7 Meses	€875.00
Assets	€25.00	Compra única	€25.00
Personal			
1 Diseñador	6.5€/hora	840 Horas	€5,460.00
1 Modelador	6.5€/hora	840 Horas	€5,460.00
1 Animador	6.5€/hora	840 Horas	€5,460.00
1 Programador	6.5€/hora	840 Horas	€5,460.00
Oficina			
1 oficina	€600.00	7 Meses	€4,200.00
Total		€30,831.65	

Tabla 4: Presupuesto Final

Preproducción

Como se comentó en la metodología del proyecto, se realizará **Preproducción**,**Producción** y **Postproducción**.

Para comenzar la **Preproducción**, lo primero que necesitamos es tener clara la historia que vamos a contar y de qué manera la adaptamos para el ámbito de los videojuegos.

La leyenda de los 47 rōnin

La leyenda de los 47 **rōnin** también conocida como el incidente de Akō (赤穂浪士, Akō rōshi), es una historia japonesa, considerada leyenda nacional se desarrolló entre 1701 y 1703 y es la leyenda más famosa que ejemplifica el código del honor samurai: el **bushido**

Asano Naganori fue escogido por el **Shōgun** para ser uno de los **Daimyo**, cuyo deber era entretener a los enviados de la familia imperial. Para ayudarlo con los protocolos de etiqueta se le asignó a **Kira Yoshinaka** el maestro de protocolos. **Kira** pensaba que **Asano** debía compensarle con una suma de dinero por el trabajo de formación que le estaba ofreciendo, pero este último creía que era el deber de **Kira** como maestro.

Ambos empezaron a tener sus diferencias y **Kira** al enterarse que no iba a recibir dinero comenzó a insultar y humillar públicamente a **Asano**.

Asano soporto las burlas durante un tiempo, además **Kira** le enseñaba erróneamente los protocolos para hacerlo quedar mal delante de la familia imperial. **Asano** comenzó a sospechar que se estaba burlando de él y al ir a pedirle explicaciones a **Kira** este volvió a humillarle, con lo que todos se rieron de **Asano**.

Finalmente **Kira** provocó tanto a **Asano** que este desenvainó su nihonto para exigirle respeto, hiriendo ligeramente en la cara a **Kira**.

Asano fue rápidamente arrestado y llevado ante la ley. A pesar de los testimonios de los testigos que aseguraban que **Kira** había provocado a **Asano**, la ley era muy estricta en cuanto a atacar a alguien en casa del **Shōgun** y esto era imperdonable.

El **Shōgun** a pesar de saber que la culpa era de **Kira**, no podía ir contra la ley y condenó de muerte a **Asano**, aunque le permitió conservar su honor y le dio la posibilidad de cometer **Seppuku** (conocido como **Hara-kiri**)

A pesar de que **Asano** había muerto, **Kira** estaba muy preocupado por los **samuráis** que estaban bajo el servicio de **Asano**, a pesar de que estos se convirtieron en **Rōnin**, decidió vigilarlos para asegurarse de que no intentarían

ningún tipo de venganza. Además incrementó el número de guardias en su casa.

Oishi Yoshio recibió un comunicado explicando la trágica historia de **Asano**, este informó a los **samuráis** ahora convertidos en **Ronin** el destino que había tenido **Asano**.

Al enterarse la mayoría se fueron buscando una vida mejor, pero **Oishi** y otros 46 se reunieron en privado para decidir qué harían ahora. A pesar de que sabían que serían severamente castigados por ello, los ahora **Rōnin** hicieron un juramento secreto para vengar la muerte de su maestro **Asano**.

Después de reunirse decidieron separarse para apaciguar todas las sospechas que pudiera tener **Kira** sobre un intento de venganza.

Oishi pasó los siguientes dos años bebiendo, apostando y acostándose con mujeres, había perdido todo el honor que le quedaba.

Al ver que el líder de los **Rōnin** estaba acabado y sin honor, **Kira** decidió despedir a muchos guardias de su casa ya que pensaba que el peligro ya había pasado.

Pero los 47 **Rōnin** que juraron vengar a su amo, solo estaban esperando el momento oportuno para llevar a cabo su plan, había estado guardando armas y creando armaduras para el gran asalto a la casa de **Kira**.

Los 47 se reunieron el decimocuarto día del decimosegundo mes de 1702, una vez en la mansión de **Kira** situada en **Edo**, decidieron asaltarla sigilosamente, pillaron a los guardias desprevenidos, estos pusieron una fuerte resistencia, pero de nada sirvió, los **Rōnin** consiguieron avanzar hasta el centro de la mansión lamentando solamente una baja.

Mientras tanto **Kira** al ver que le estaban atacando, intentó esconderse con las damas de la casa.

Después de combatir con todos los guardias, los **Rōnin** investigaron toda la casa para finalmente encontrarse con **Kira** el cual fingió ser un sirviente, pero **Oishi** lo reconoció.

A pesar de todo el esfuerzo que hicieron para vengarse, los **Rōnin** seguían fieles a su amo **Asano** y le dieron la oportunidad a **Kira** de morir de una forma honorable, cometiendo **Seppuku**.

Kira desprecia a los **Rōnin** y los insulta, **Oishi** toma esto como un no y ordena que pongan de rodillas a **Kira** y lo decapita.

Los **Rōnin** envuelven la cabeza en una tela y la llevan al templo **Sengakuji** donde estaba enterrado **Asano**. Allí se la presentan a su amo completando finalmente su venganza.

Varios soldados llegan para arrestar a los **Rōnin** que pacíficamente se entregan ya que no tienen motivos para luchar, aceptarían su castigo sin dudarlo.

El **Shōgun** les aplica la ley, matar a un **Daimyo** se penaliza con la muerte. Pero al haber llevado a cabo tal muestra de lealtad se les permite morir de forma honorable, practicando el **Seppuku**.

Los 47 **Rōnin** son enterrados junto a su amo y a dia de hoy se puede visitar sus tumbas en el templo **Sengakuji** en **Tokio**.

Adaptación para Bushidō

La adaptación de la leyenda de los 47 **Rōnin** para este proyecto se basa en centrar la acción en el asalto a la mansión de Kira. En lugar de tener a 47 guerreros, solo controlaremos a Oishi en una misión de infiltración que acabará en una masacre en solitario.

La acción se dividirá en 3 zonas.

- Zona de la Muralla.
- Zona de la Plaza
- Zona de la Montaña.

Más adelante veremos cómo se desarrolla la acción a través del mapa con el **Level Design Document** o **LDD**

MoodBoard

Ahora que ya sabemos cual es la trama de nuestro trabajo, necesitamos llevarlo a cabo y para esto debemos hacer un trabajo de investigación y referencias para poder realizar un correcto trabajo de diseño de nivel y estética.

Como hemos comentado antes, la trama de nuestro proyecto transcurre durante los años 1701 y 1703, lo cual equivale al periodo Edo de la historia Japonesa. Un periodo marcado por la paz que se estableció por todo Japón debido a la lealtad que los Daimyos al Shōgun,

Al tratarse de un escenario nocturno se ha hecho un **moodboard** acorde a este además de añadir elementos diurnos para poder distinguirlos mejor.

Fig. 25 MoodBoard

Level Design Document

Para diseñar el Escenario del proyecto, nos hemos basado en una mansión japonesa de la época. La cual cuenta con una muralla exterior que rodea todo el escenario.

Tiene un pequeño lago en la entrada con un puente que lo atraviesa, seguido de un pequeño patio el cual conduce hasta la entrada principal de la mansión. En la parte trasera de la casa se encuentra una pequeña meseta en la cual hay un templo budista.

Como hemos comentado antes, el nivel está dividido en 3 zonas.

Muralla

La muralla es la primera parte del mapa, en la cual el jugador llegará y se encontrará con el primer **QTE**. El combate a muerte contra un samurai al cual le tocaba hacer guardia. El enemigo realizará un ataque, el cual el personaje o bien bloqueara para realizar un contraataque letal o bien morirá.

Si el jugador sale victorioso, se adentrará en la muralla del castillo, abriendo un portón el cual es el segundo **QTE**

Plaza

Para llegar a la plaza el jugador deberá cruzar un pequeño lago a través del puente, en el cual visualiza a un segundo guardia esperando para otro combate a muerte. Aqui esta el tercer QTE. Una vez acabado el combate, el jugador verá una pequeña cinemática donde Kira saldrá corriendo desde la mansión equipado con una katana en dirección a la montaña.

El jugador deberá seguir a Kira hasta lo alto de la montaña.

Montaña

La Montaña es la última parte del mapa, En la cual el jugador luchará contra Kira. Tiene una ligera cuesta que el jugador deberá subir para llegar a la meseta donde se encuentra un pequeño jardín con un templo budista al final de este.

Una vez llegue a la meseta, se reproducirá una cinemática con 3 QTE en los cuales se decidirá el resultado del combate a muerte entre Oishi y Kira.

Sketch del Mapa

Ahora que tenemos definidas las 4 zonas del mapa toca hacer una primera iteración sobre el papel, la cual nos permitirá definir distancias y áreas de mapa jugable.

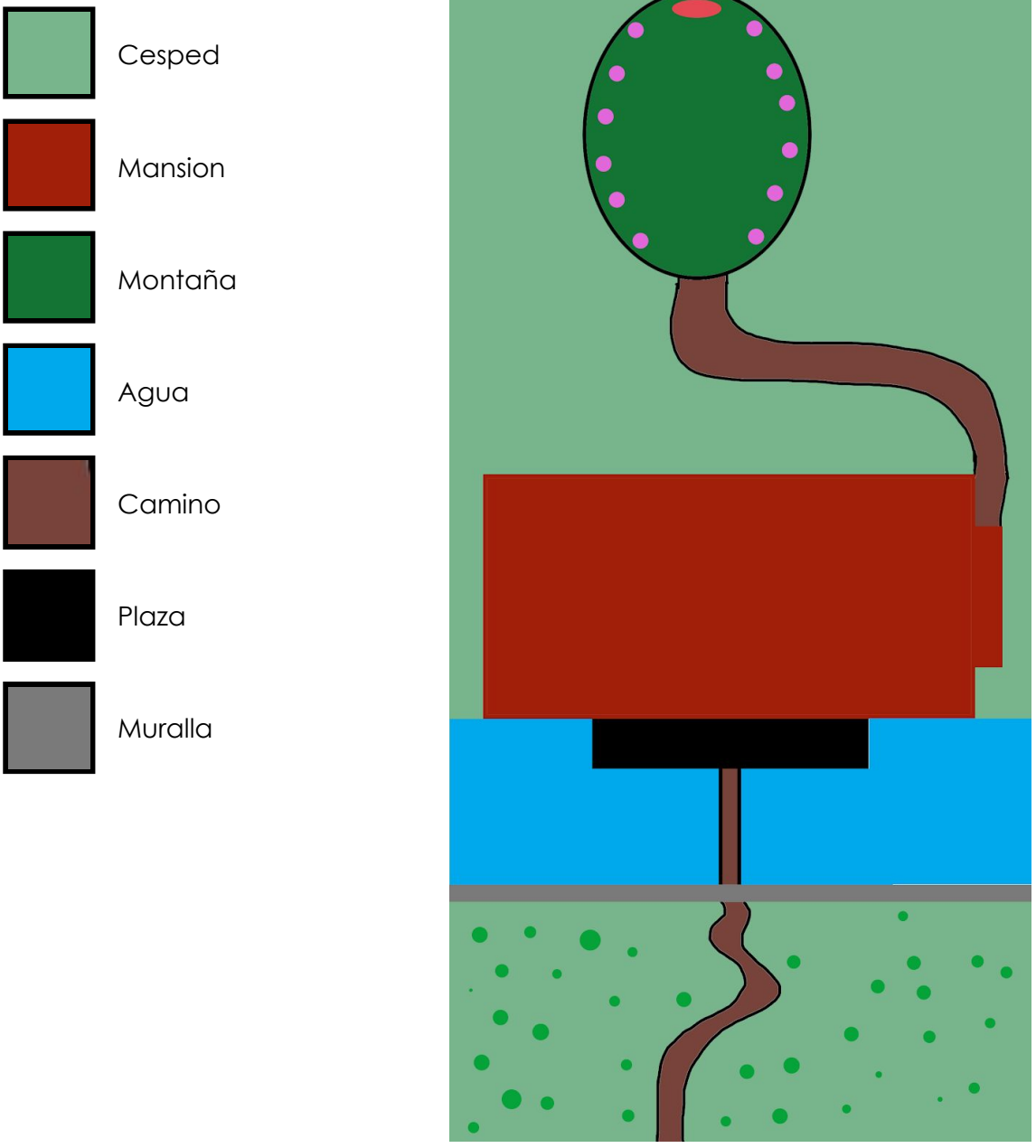


Fig. 26 Sketch del Mapa

Primera Iteracion del mapa

Después de realizar el **sketch** se realiza otra versión del mapa para que sean más claras las zonas y que sea más fácil moverse por él. Al tratarse de una mansión del **Daimyo**, debe estar totalmente fortificada.

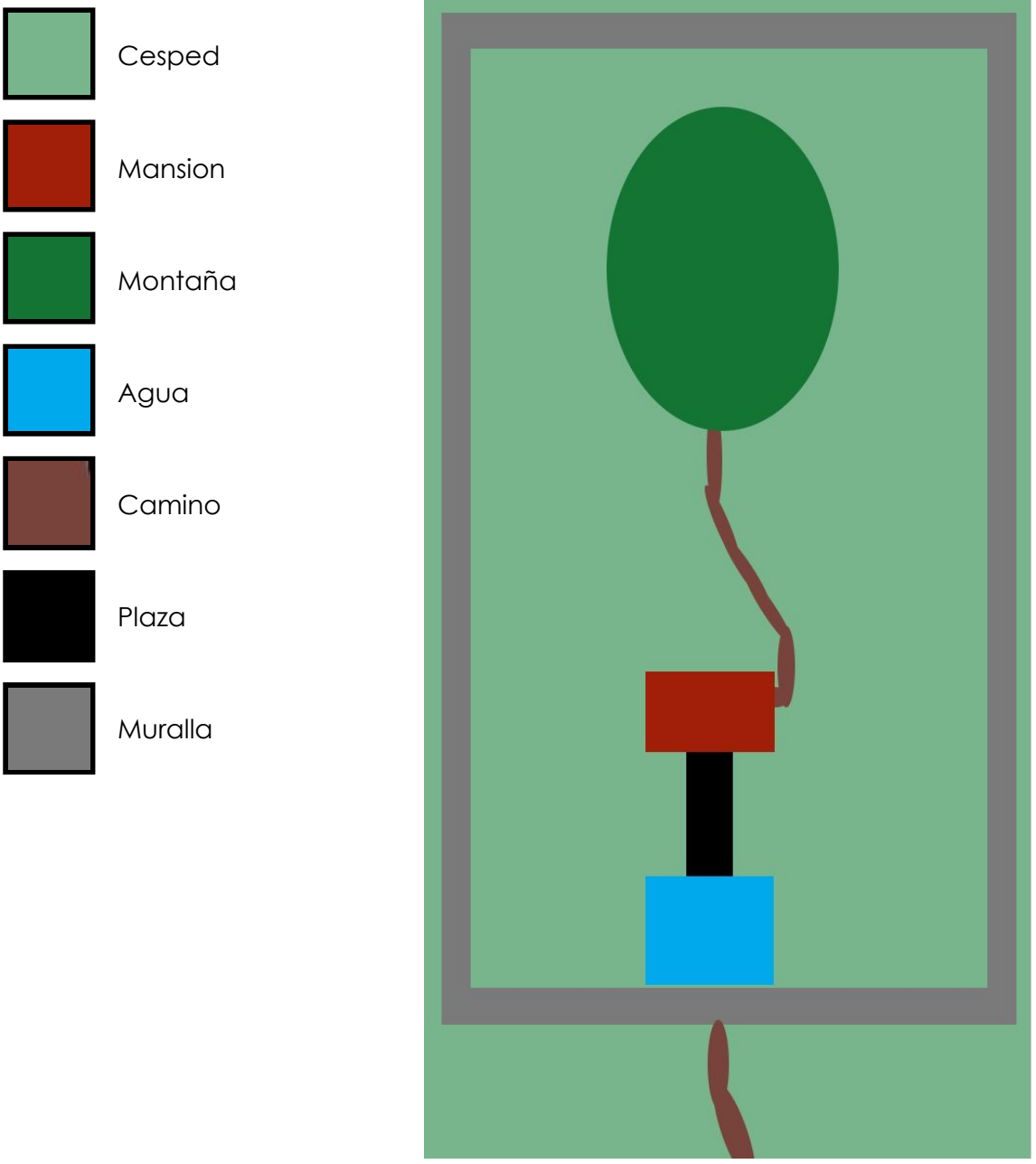


Fig. 27 Primera iteración del mapa

StoryBoard

Para tener claro el guión que va a seguir nuestro proyecto necesitamos plasmarlo en papel. Para esto utilizaremos un **Storyboard** también conocido como Guión gráfico.

Un **Storyboard** es una sucesión de ilustraciones o imágenes que sirven para previsualizar una animación, una película o cualquier tipo de contenido audiovisual. El Storyboard tal como lo conocemos a día de hoy fue desarrollado por en Walt Disney Productions durante los años 30, a pesar de esto, otros estudios de animación utilizaban técnicas similares.

En este proyecto se utiliza un Storyboard de 6 ilustraciones con una breve descripción de la acción a realizar, el número de escena y si en este momento hay un QTE activo.

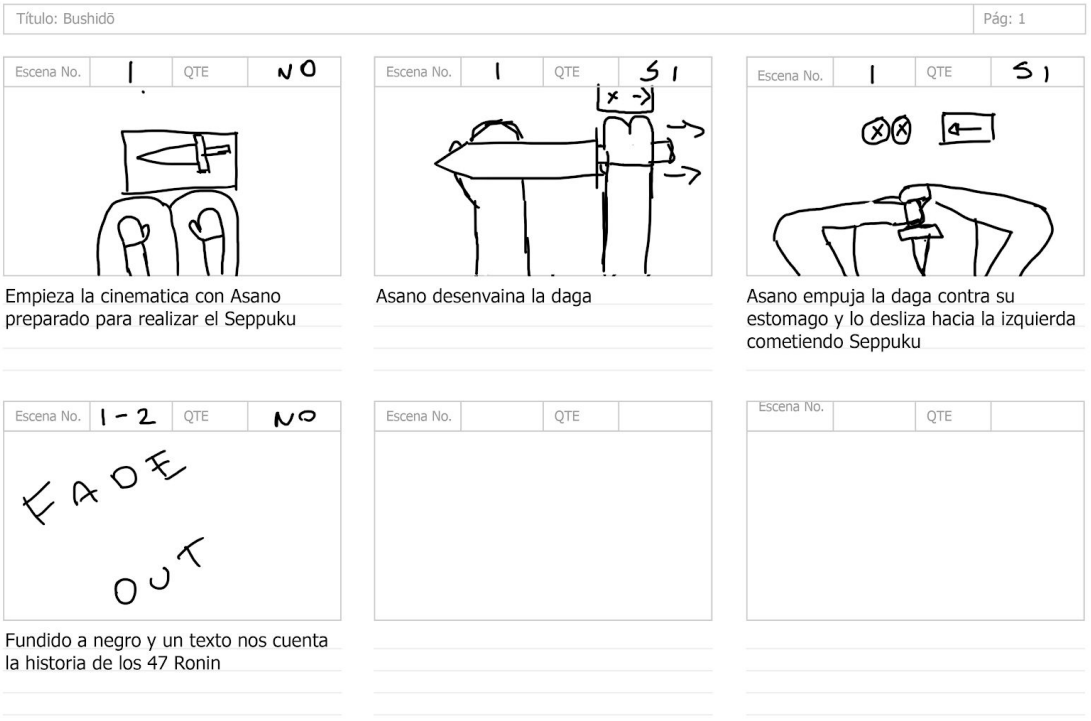


Fig. 28 Primera página del storyboard

El resto del **Storyboard** se puede consultar en Anexo

Personajes

Este proyecto cuenta con 3 personajes principales y uno secundario.

Personajes Principales

Oishi Yoshio

Personaje principal del proyecto y al cual el jugador controlará. Líder de los 47 **Rōnin** y un **samurai** impecable.

Kira Yoshinaka

Daimyo de **Edo**, principal responsable de la muerte de Asano. Jefe final del juego y un experto **samurai**.

Asano Naganori

Amo de los **Rōnin** y condenado a muerte por el Shogun por intentar matar a **Kira**.

Fig. 29 Modelo de Oishi

Fig. 30 Modelo de Kira

Fig. 31 Modelo de Asano

Personajes Secundarios

Guardia Samurai

Guardia de la plaza, realiza ataques desde todas direcciones además de una estocada.

Fig. 32 Samurai

Listado de animaciones

Una vez tenemos el **storyboard** y los personajes que van a salir, necesitamos tener un listado de animaciones para producir.

Para esto utilizamos una hoja de cálculo de **Google Drive** para tener constancia y separar por personajes las animaciones, además de reflejar en qué parte del proceso de animación se encuentran.

Link
https://docs.google.com/spreadsheets/d/1TiakHXOml0dCpg3jdZlGtIphgG6K_ePf8qne3Fxkgxk/edit?usp=sharing

fx | Lista de Animaciones

	A	B	C	D	E	F	G
1	**Lista de Animaciones**						
2							
3							
4	**Personaje Principal**	Referencia	Blocking	Spline	Polish	In-game	
5							
6	Walk	X					
7	Run	X					
8	crouch Walk	X					
9	Idle	X					
10	RuntoWalk	X					
11	RunToIdle	X					
12	Unsheathe	X					
13	GetHit	X					
14	Die	X					
15	Combat Pose 1	X					
16	Combat Pose 2	X					
17	Jump 1	X					
18	Jump 2	X					
19	Falling	X					
20	Block Left	X					
21	Block Right	X					
22	Block Up	X					
23	Block Thrust	X					
24	Attack1	X					
25	Attack2	X					
26	Attack3	X					
27	Finisher1	X					
28	Finisher2	X					
29	Finisher3	X					
30							
31							
32	**Enemigo Guardia Lanza**						
33							
34	Idle	X					
35	Walk	X					
36	GetHit	X					
37	Attack1	X					
38	Attack2	X					
39	Die	X					
40							
41							
42	**Enemigo Samurai**						
43							
44	Idle	X					
45	Walk	X					

Fig. 33 Listado de Animaciones

Producción

En la parte de **Producción** realizaremos todo lo necesario para llevar a cabo el proyecto dentro de nuestro motor de videojuegos.

White boxing

Después de haber realizado el concepto del mapa, procedemos a realizar un **White Boxing** que consiste en utilizar un motor de videojuegos para poner "cajas blancas" que den forma al mapa, esto se utiliza para poder explorar el mapa y realizar cambios de forma, tamaños, áreas o incluso cambiar el mapa por completo si no cumple con las expectativas.

Esto se utiliza a nivel profesional porque es seguro, es decir, una vez el white boxing está acabado, sabemos que este funciona, que sus proporciones son buenas, que su flow es el correcto y que funcionará sea un mapa del desierto, uno nevado o un mundo post apocalíptico porque el diseño de este es el adecuado.

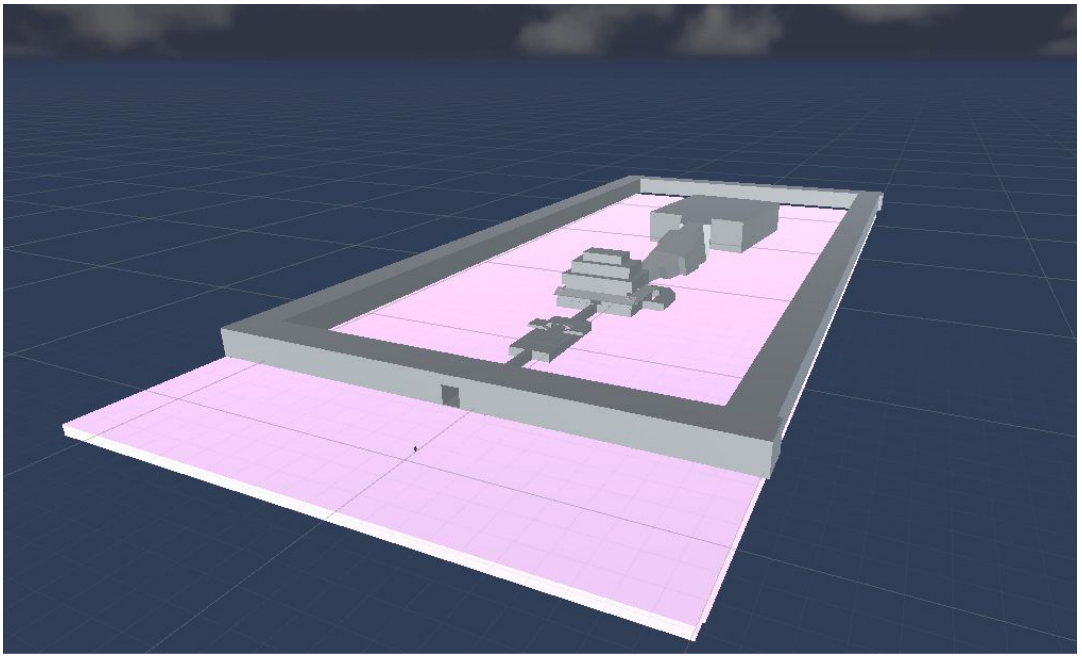


Fig. 34 White boxing del nivel en Unity

Desarrollo del Mapa

Una vez realizado el White boxing y vemos que funciona bien, se pasa a poner los assets en su sitio y darle temática y vida al mapa.

A continuación veremos las comparaciones entre el **white boxing** y la fase final del mapa.

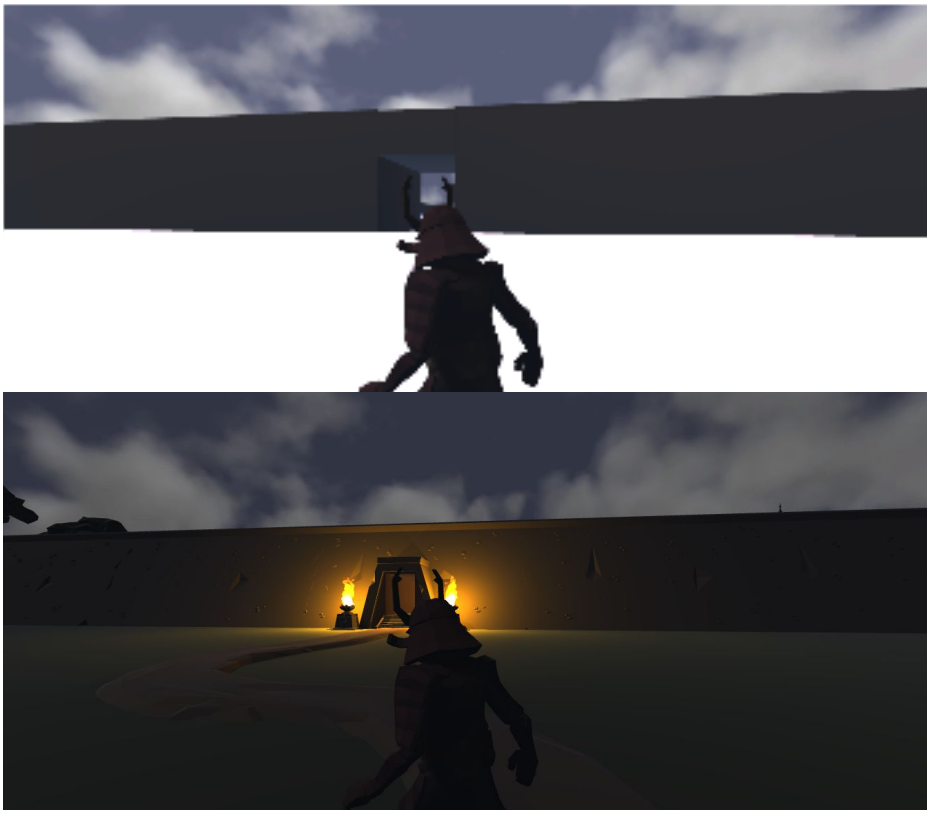


Fig. 35 Comparación 1 desarrollo con white boxing

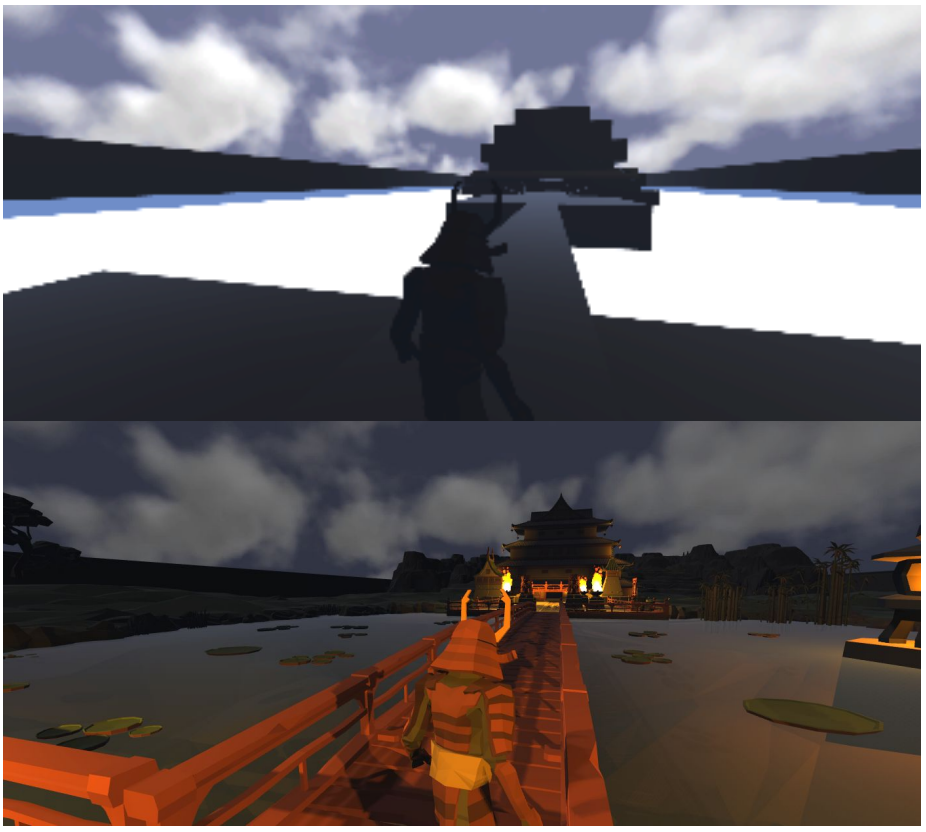


Fig. 36 Comparación 2 desarrollo con white boxing

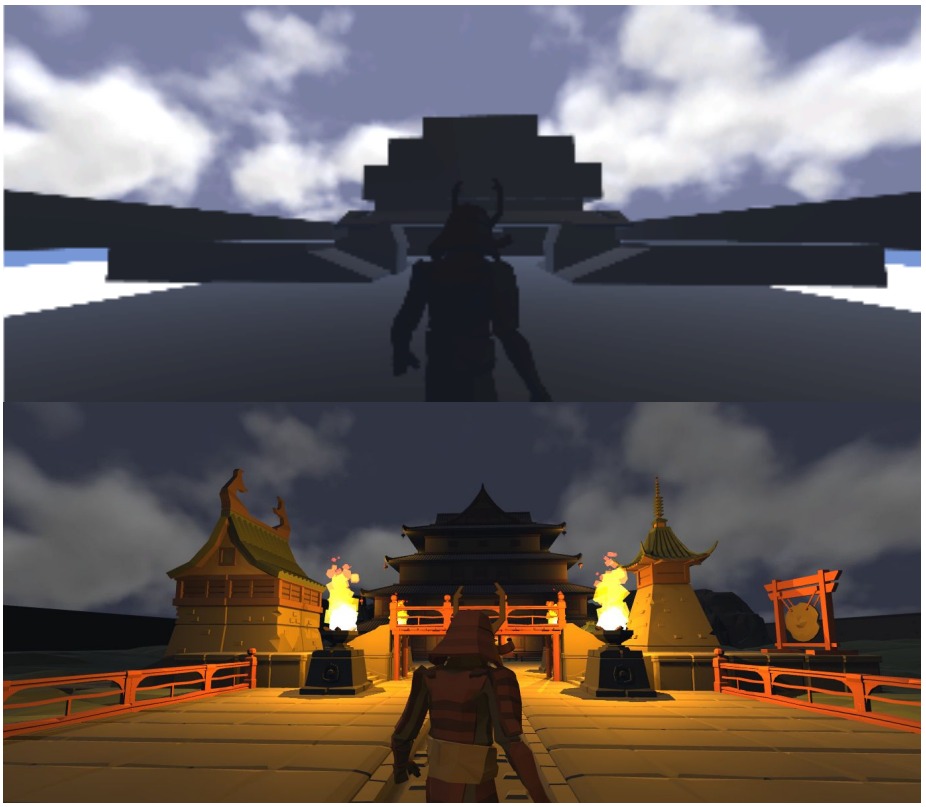


Fig. 37 Comparación 3 desarrollo con white boxing

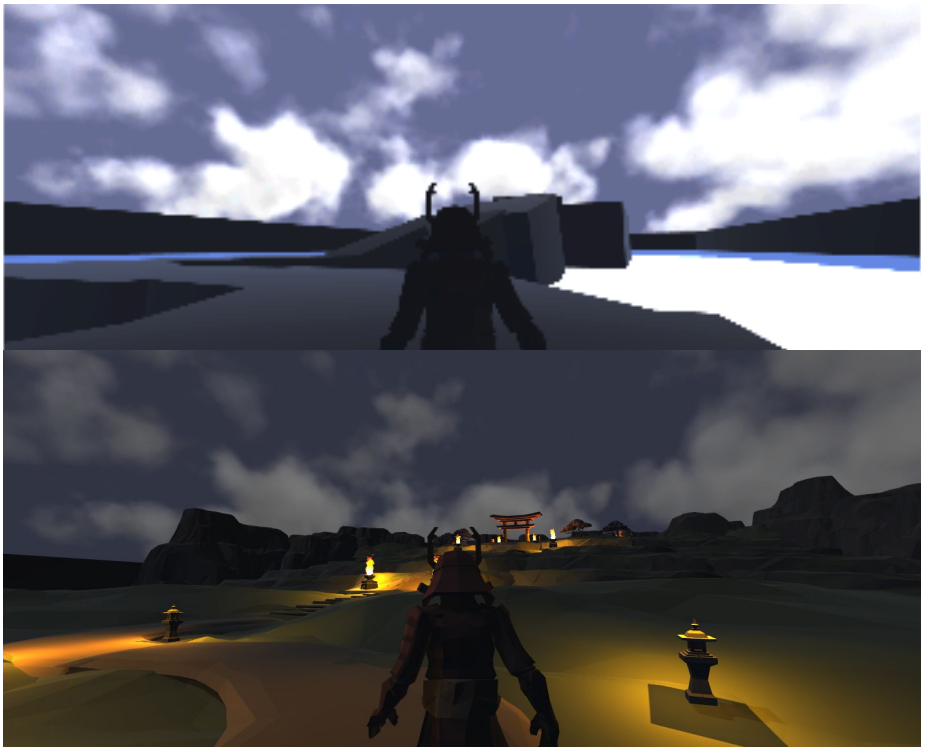


Fig. 38 Comparación 4 desarrollo con white boxing

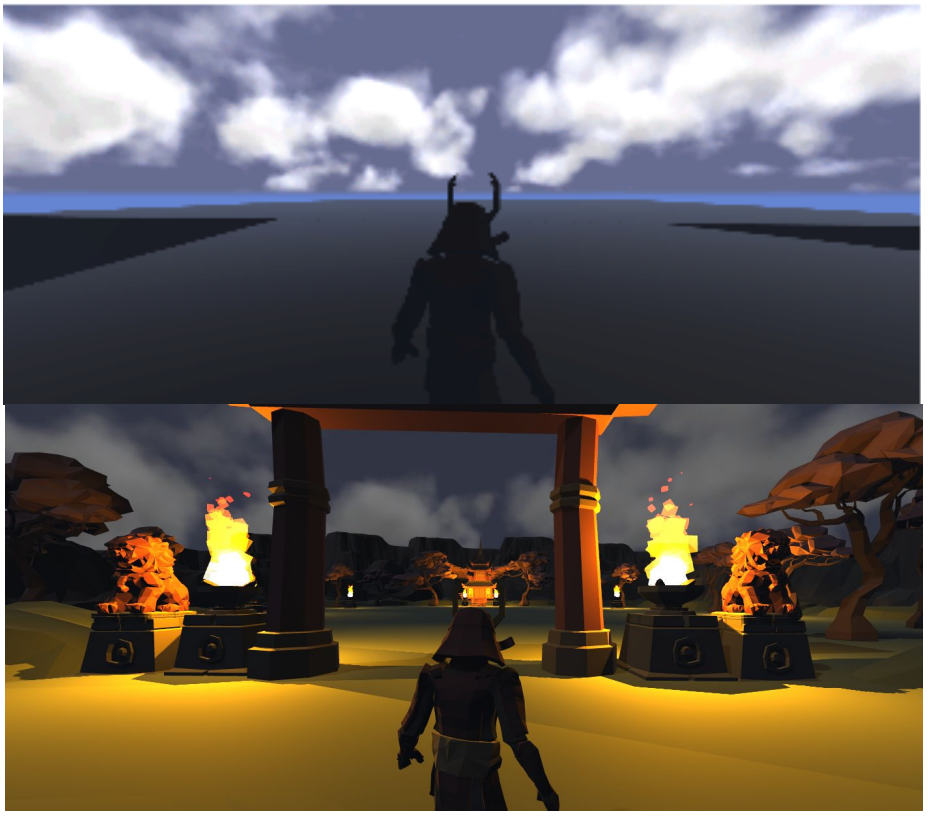


Fig. 39 Comparación 5 desarrollo con white boxing

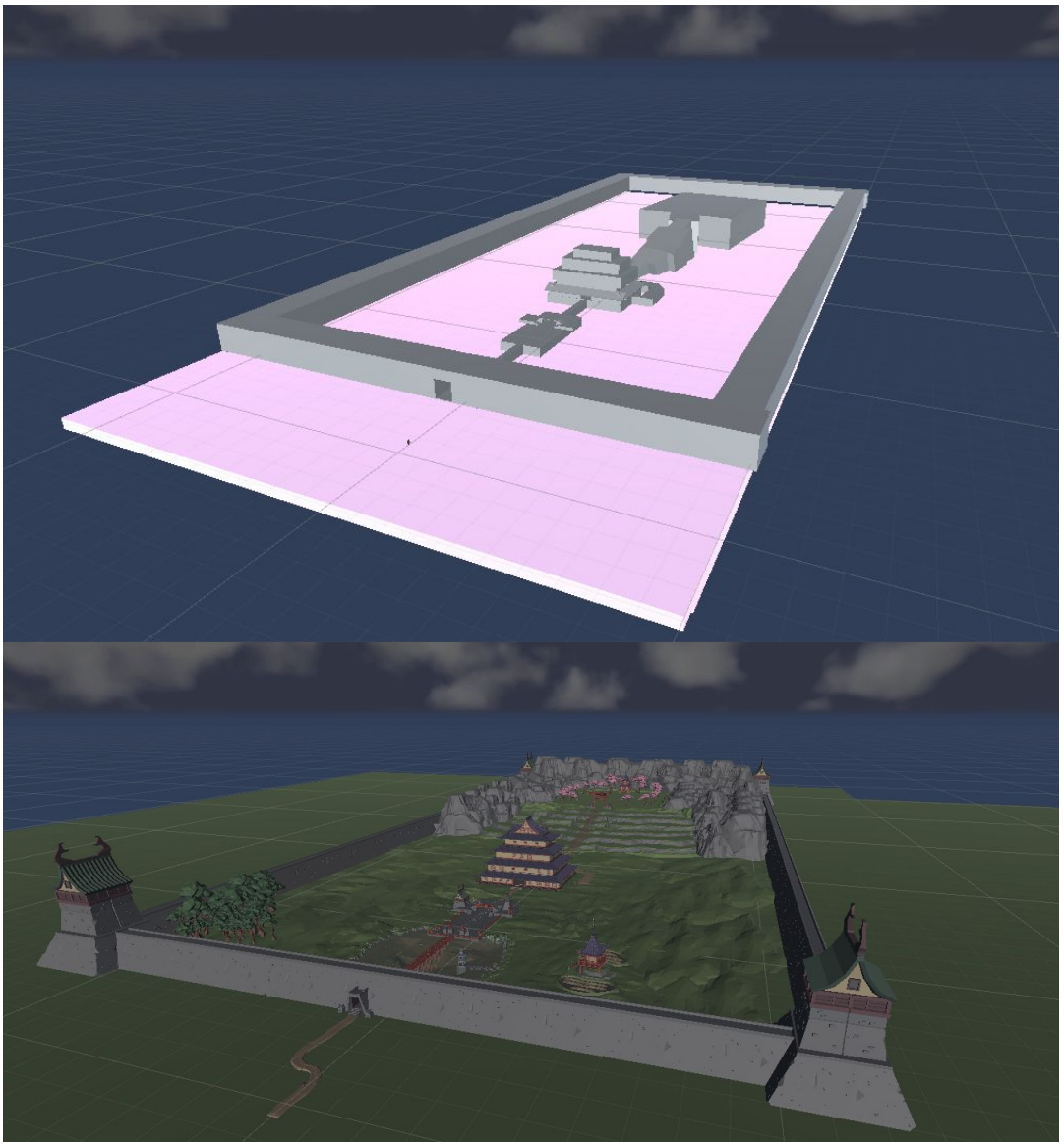


Fig. 40 Comparación 6 desarrollo con white boxing

Una vez hemos concluido de construir nuestro mapa en **Unity**. es hora de preparar nuestros personajes para realizar las animaciones.

Preparación de Personajes

Como hemos comentado anteriormente, hemos comprado **Assets** para nuestro proyecto, en los cuales se incluyen personajes con **Rig** y **Skinning** que aprovecharemos.

Como todos los personajes comparten el mismo esqueleto, solo hace falta realizar controladores y las adaptaciones que necesitamos solo a un modelo 3D. En este caso seleccionaremos el modelo de nuestro personaje principal **Oishi**.

Fig. 41 Modelo Personaje principal y Rig

Joints y controladores

Como hemos comentado anteriormente, para poder animar a un personaje, necesitamos que este disponga de un esqueleto.

Este se crea utilizando una jerarquía de Joints, es decir, de un Joint principal, se van creando joints que derivan (o son hijos) de este.

Normalmente en personajes bipedos y quadrupedos este joint principal se encuentra en la cadera, aunque hay veces que se crea un joint anclado al suelo que funciona como principal y del cual sale el joint de la cadera.

Fig. 42 Jerarquía de joints standard

La rotación y transformación de un joint hijo la heredó de su predecesor, es decir que si movemos el joint central "Hip" 10 cm hacia la izquierda, todos los joints de su jerarquía se moverán 10 cm hacia la izquierda, de igual manera ocurre con las rotaciones.

Ahora se crean los controladores que nos permitirán mover los huesos del personaje. Utilizando el software **Maya** y la herramienta de creación Nurb Curves, creamos las formas geométricas que nos servirán como controladores.

Estos pueden tener la forma que más nos agrade, pero se acostumbran a utilizar circunferencias y cajas.

Se crea un controlador por cada joint que haya en la jerarquía, exceptuando los joints extremos en algunos casos, es decir los últimos de la jerarquía.

Los controladores se utilizan para realizar las rotaciones y traslaciones que queremos que hagan los joints, para ello se necesita una conexión entre la transformación del joint y la de su controlador. Podríamos hacer que cada controlador tenga como hijo al joint que le corresponde, pero hay veces que no interesa que la traslación o la rotación de un controlador se traslade al joint, o simplemente queremos desactivarla durante un tiempo para realizar otras acciones con conecciones especiales que veremos más adelante.

Para ello se utilizan los constraint de maya.

Constraints

Los constraints son tal como su nombre indica, restricciones que un objeto impone sobre otro independientemente de si es su padre o no.

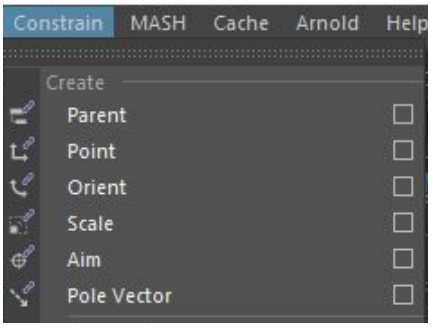


Fig. 43 Constraints dentro de Maya

Existen diferentes tipos de Constraint dentro de maya:

Parent Constraint: Con un parent constraint puedes relacionar las rotaciones y translaciones de un objeto a otro, como si una coneccion Padre-hijo de jerarquía se tratara. Más adelante se detallan las diferencias entre un Parent Constraint y un Parent.

Point Constraint: Con un Point constraint relacionas las traslaciones de un objeto a otro.

Orient Constraint: Con un Orient Constraint relacionas los atributos de rotación de un objeto a otro.

Scale Constraint: Con un Scale Constraint relacionas los atributos de escala de un objeto a otro.

Aim Constraint: Con un Aim Constraint consigues que un objeto "mire" a otro, esto es muy útil para los controladores de los ojos de los personajes.

Pole Vector Constraint: Con un Pole Vector Constraint consigues que un Pole Vector de IK se mueva y siga la posición de otro objeto

Una vez tengamos todos los controladores creados, nos disponemos a enlazarlos cada uno con su **Joint** a través del parent constraint.

Diferencias entre Parent y Parent Constraint

Como se ha estipulado antes, las conecciones entre los Joints y los controladores se realizan a través de un Parent Constraint(PC), que a simple vista funciona igual que un Parent pero si analizamos a fondo encontramos muchas diferencias.

Jerarquía: La primera diferencia que encontramos está dentro de la jerarquía, si utilizamos un Parent el Joint pasa a ser hijo del Controlador, mientras que si utilizamos PC la jerarquía no se ve afectada.

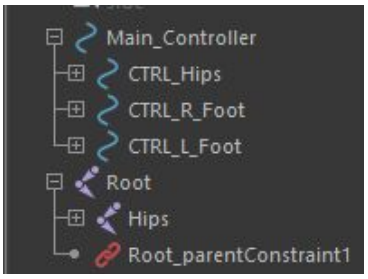


Fig. 44 Jerarquia Parent Constraint

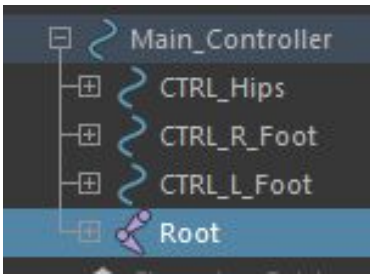


Fig. 45 Jerarquia Parent

Activar o Desactivar conexión: Otra diferencia es la opción de activar y desactivar la conexión entre ambos, en el caso del Parent no se puede desactivar mientras que en PC sí.

Orden de selección: El orden de selección para crear estas conexiones es diferente, para crear un Parent primero seleccionamos al hijo, mientras que para un PA primero seleccionamos al padre.

Libertad de movimiento: Cuando realizamos un Parent, el hijo de este tiene total libertad de movimiento, simplemente que esta estará guiada por la de su padre, mientras que en un PA el movimiento está restringido y se señala en maya con el color azul.

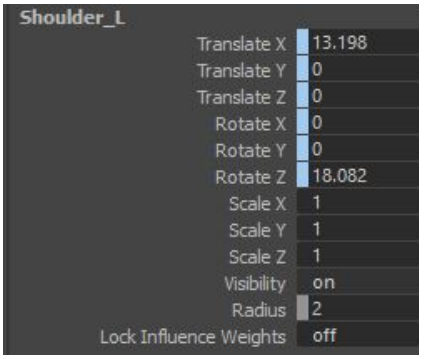


Fig. 46 Atributos bloqueados por Parent Constraint

Control de la escala: con ambas conexiones se puede modificar tanto la rotación como la traslación del objeto hijo, pero la diferencia es que en un Parent puedes también cambiar el escalado de este.

Atributos afectados: Un Parent no tiene opciones de cómo afectará a su hijo, es o todo o nada, mientras que un PA tienes diferentes opciones, como afectar solo unos ejes en específico o cambiar el porcentaje que afectará la conexión a los atributos del hijo, con un valor entre 0 y 1.

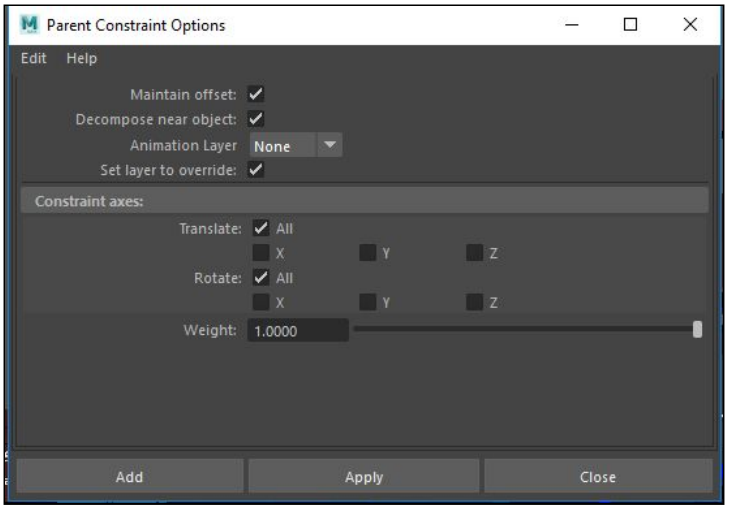


Fig. 47 Opciones de Parent Constraint

Key Frames: En una conexión Parent no puedes realizar Keyframes en la relación padre-hijo, simplemente puedes hacerlo independientemente de su conexión, mientras que en un PA cuando realizas un Key Frame en el hijo, aparece una opción llamad "Blend Parent" que te permite activar o desactivar el Constraint.

Debido a estas diferencias, a la hora de animar se utiliza un Parent Constraint para conectar los Controladores a los Joints. Ahora bien, aquí se ha detectado un problema con los ejes de rotación de los controladores, los cuales se alinean con los ejes del mundo y si queremos rotar un **joint** que no esté orientado con este, como por ejemplo brazos y dedos, nos modificará más de un eje de rotación cuando lo que queremos es que roten de manera limpia, es decir que solo se modifique un eje de rotación.

Para solucionar este problema lo que se ha hecho es entre cada controlador y su hijo crear un grupo, el cual será el encargado de recibir las rotaciones del mundo, mientras que el controlador rotará con referencia al **joint**.

Para alinear los **controladores** al **joint**, se ha utilizado un script desarrollado por *Serge Scherbakov* **Technical Artist** en Jagex de animadores que lo hace automáticamente.

Script:http://www.serge-scherbakov.com/2012/12/align-pivots.html

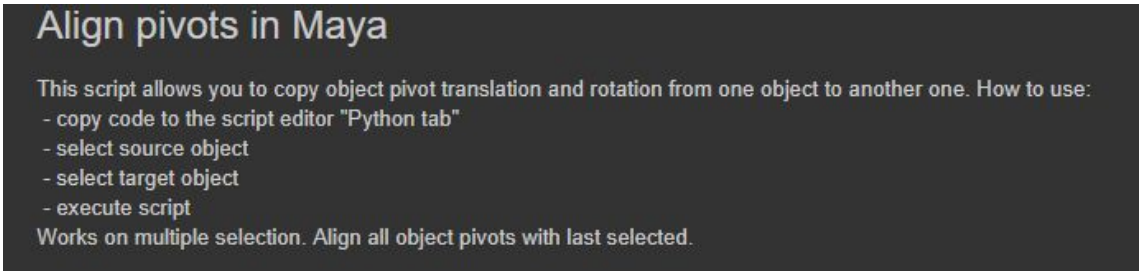


Fig. 48 Instrucciones para utilizar Script

Las siguientes 4 imágenes muestran el proceso en el siguiente orden:

Imagen 49: Eje de rotación del **Joint** (el cual no está alineado al mundo)

Imagen 50: Eje de rotación del **controlador** (el cual está alineado al mundo)

Imagen 51: Eje de rotación del **grupo** con sus atributos de rotación editados

Imagen 52: Eje de rotación del **controlador** siendo hijo del grupo, con sus atributos de rotación intactos.

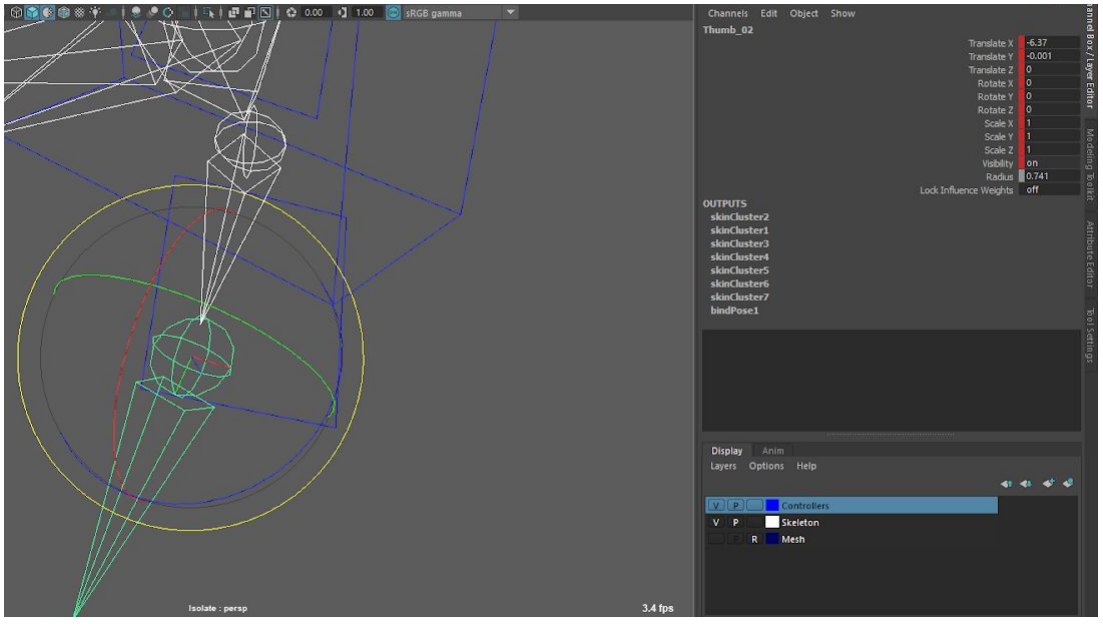


Fig. 49 Joint y su eje de rotación

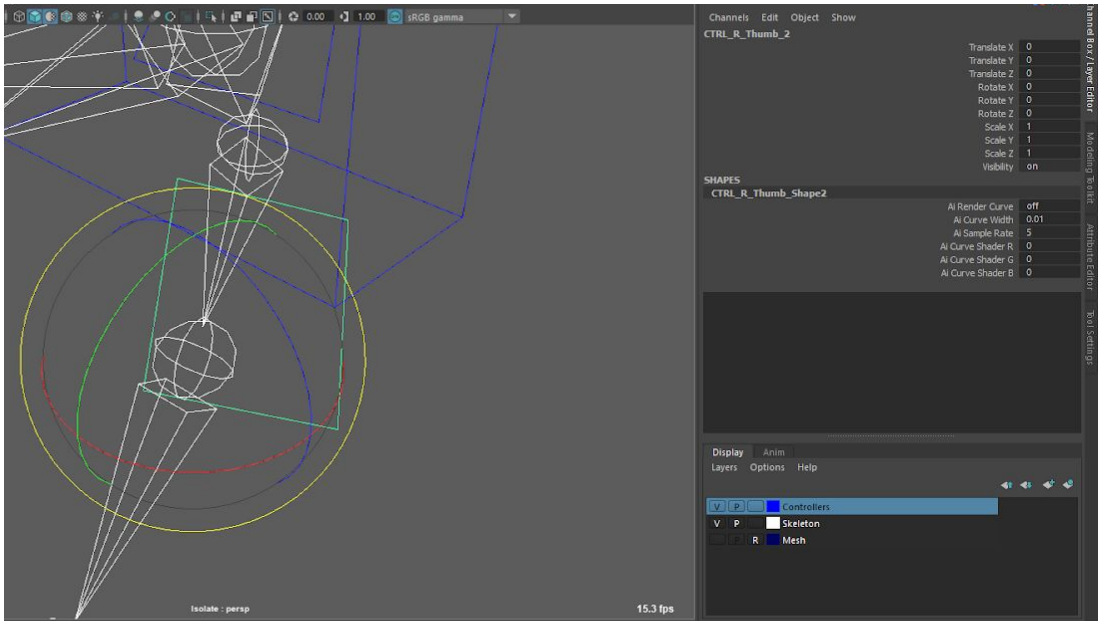


Fig. 50 Controlador y su eje de rotación

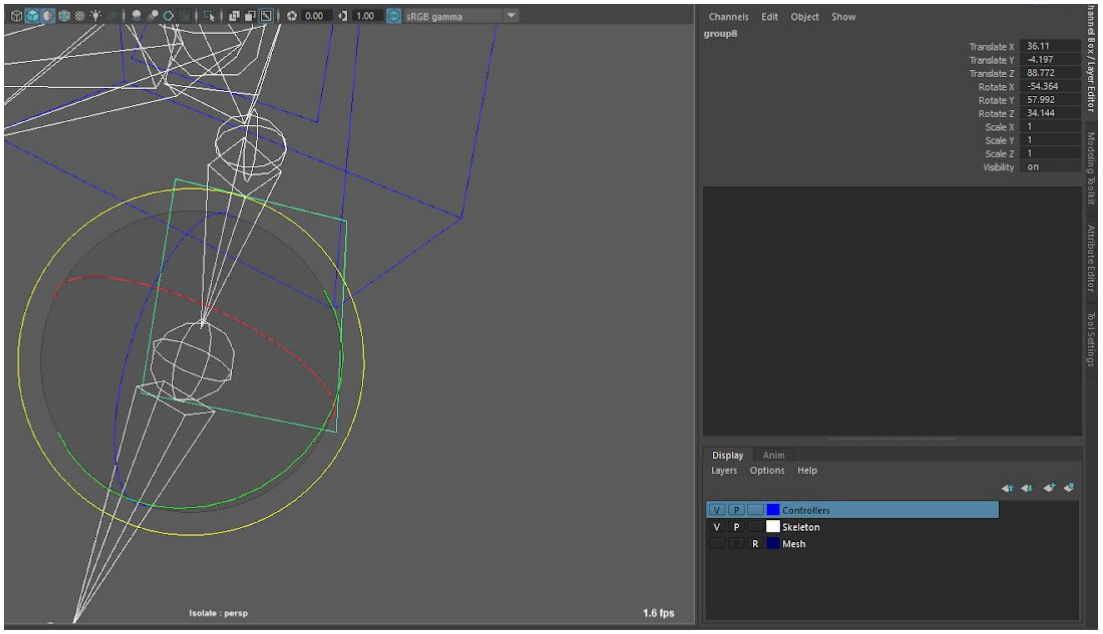


Fig. 51 Grupo y su eje de rotación

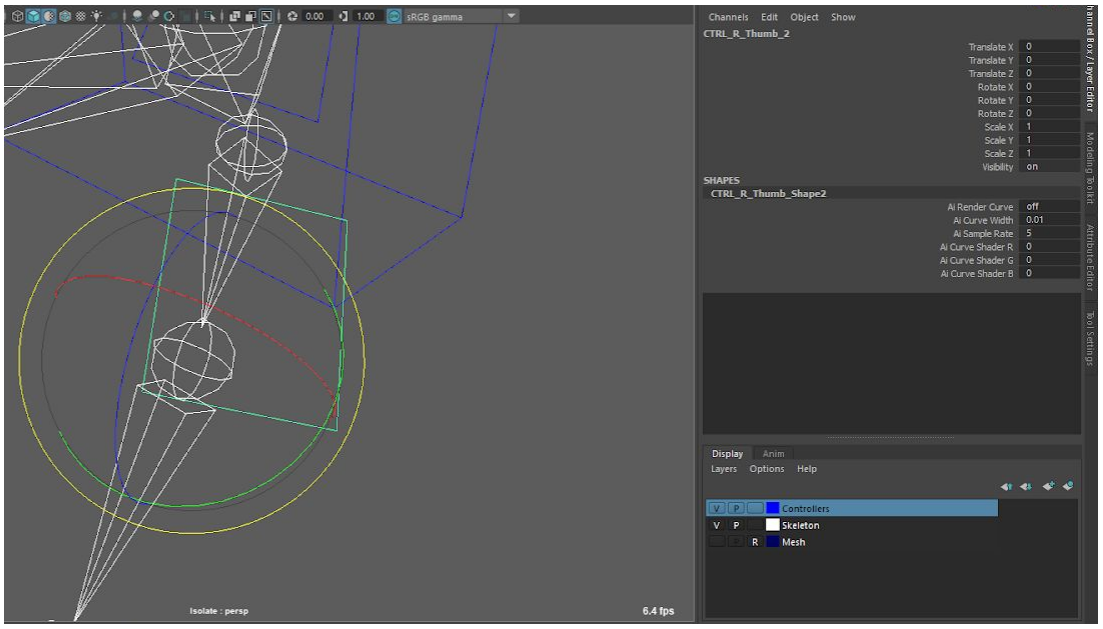


Fig. 52 Controlador y su eje de rotación

Como vemos en la imagen 49 y en la imagen 52 ambos ejes de rotación coinciden y están inicializados a 0 y preparados para animar.

Además de los modificadores anteriores se ha utilizado **Inverse Kinematics (IK)**. Inverse Kinematics es el proceso por el cual una cadena de Joints calcula su posición automáticamente solo importando donde el último joint de la cadena está situado.

Al contrario de Forward Kinematics **(FK)** que es el método utilizado normalmente en la animación, en **IK** el Joint hijo es el que modifica la rotación de su padre mientras que en **FK** es el Joint padre el que modifica al hijo.

IK se utiliza cuando nos importa más dónde estará el extremo de una cadena de joints qué en qué posición estarán los Joints padres. Un ejemplo muy claro son los Joints de las piernas.

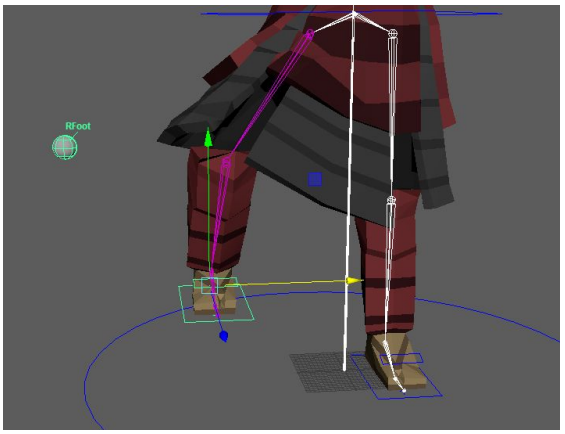


Fig. 53 Inverse Kinematics

Aquí es donde se utiliza el constraint de Pole Vector, este constraint nos permite modificar la rotación de la cadena de Joints, para esto se crea un Locator o un objeto con el cual enlazaremos el Constraint a la cadena de Joints.

Este modificador además de las piernas se ha utilizado en los brazos para realizar la animación de "Abrir el Portón".

Una vez tenemos los **controladores** necesarios ya podemos comenzar a producir nuestras animaciones.

Fig. 54 Modelo con controladores

Producción de Animaciones

Una vez está completo el **Listado de Animaciones** y tenemos preparado al personaje para ser animado, procedemos a seguir la pipeline de animación.

Referencias

Lo primero que tenemos que hacer para realizar una animación, es capturar referencias.

Para esto simplemente tenemos que grabar a personas realizando la acción que queremos animar para así tener una guia de como se comporta el cuerpo en el mundo real, a pesar de esto, luego se aplicaran principios de animación para realizar una correcta animación.

Fig. 55 Fotograma de las referencias

Para capturar las referencias se han utilizado dos cámaras deportivas estilo "GoPro" sujetadas en dos trípodes que se localizan en el frente y el lateral de la acción para capturar el mayor ángulo posible y tener una vista de todas las partes del cuerpo y que no se pierda ningún detalle.

Una vez tenemos las referencias grabas es hora de prepararlas para poder utilizarlas. Para esto utilizamos el programa "Adobe Premiere Pro" en este recortamos los videos y quedarnos solamente con la acción que nos interesa, después sincronizamos los dos videos, como si de una película se tratara, para esto durante la grabación se puede utilizar una claqueta, o de una forma más casera dar una palmada. Una vez los tenemos sincronizados y recortados, los exportamos a una resolución de 1080p y a 30fps.

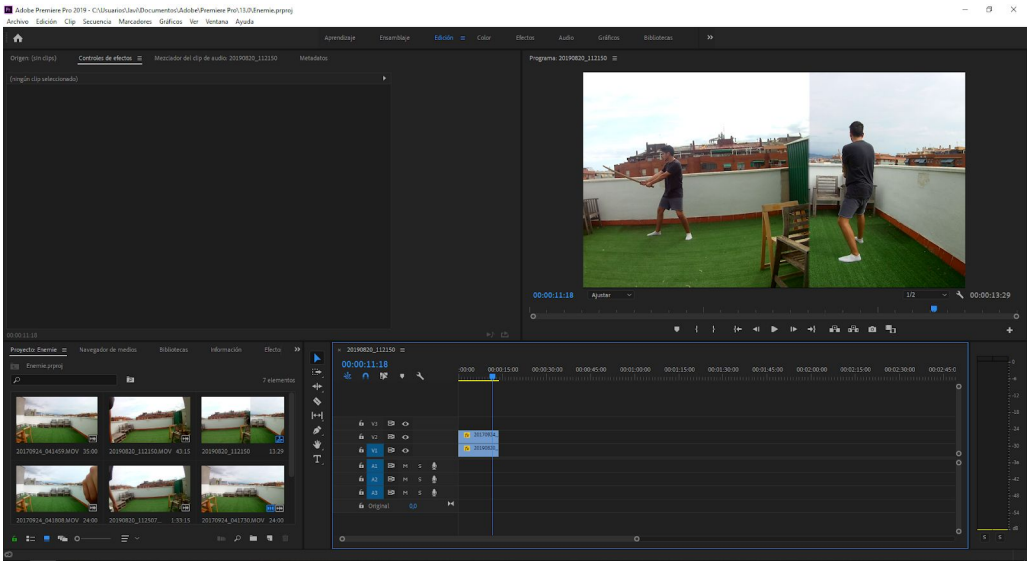


Fig. 56 Adobe Premiere con Referencias

Para reproducir los videos de referencia a la vez que utilizamos maya se ha comprado un software **Third Partner** llamado **KeyFrame MP2**, este programa nos permite sincronizar la línea de tiempo de maya con la reproducción del video de referencia. Este software está desarrollado por Chris Zurbrigg y tiene un precio de 33.00€ que añadimos a nuestro presupuesto.

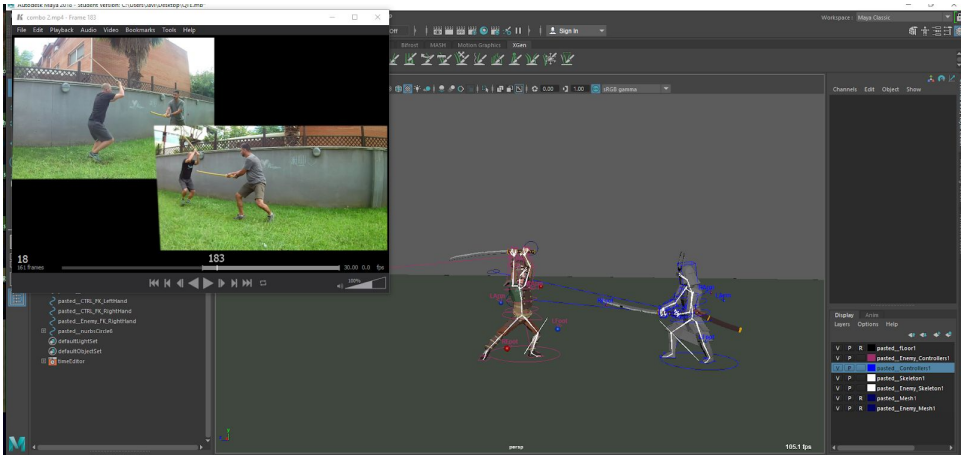


Fig. 57 KeyFrame MP2 junto a Maya

Proceso de Animación

Para crear las animaciones se sigue el proceso Pose to Pose, el cual se basa en realizar poses de fotogramas claves o también conocidos como "**Key Frame**" para ello seleccionaremos los fotogramas que más peso tengan en el movimiento. Comenzaremos por la animación más básica de todas donde se verán claramente los ejemplos de **Key Frame**.

Walk Cycle

Para animar un Walk Cycle nos basamos en el libro "The Animator's Survival kit" en el cual nos enseña los Keyframe de manera clara.

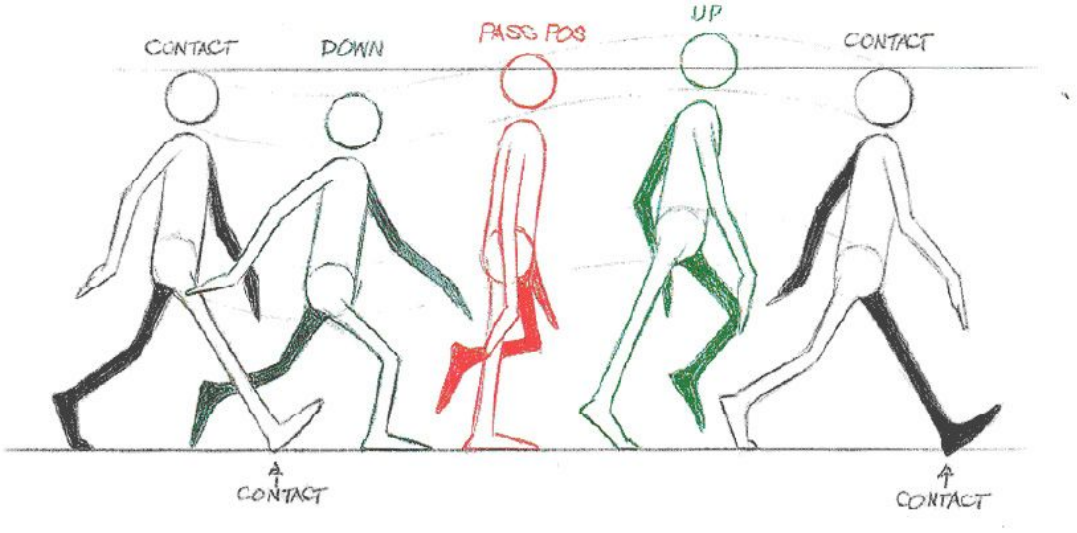


Fig. 58 Walk Cycle Animator's Survival Kit

Fig. 59 Walk Cycle Bushido

Una vez tenemos todos nuestros fotogramas claves, realizamos los In-Between que son fotogramas entre los fotogramas clave, estos nos ayudan a darle ritmo a la animación y detallar que arcos seguirán los brazos y piernas de nuestro personaje.

Una vez finalizado la fase de blocking pasamos a activar las interpolaciones entre **Key Frames**

Fase de Spline

En esta fase ya podemos visualizar el movimiento total de nuestro personaje de forma fluida, en esta fase nos basaremos en retocar las curvas de interpolación para corregir pequeños errores.

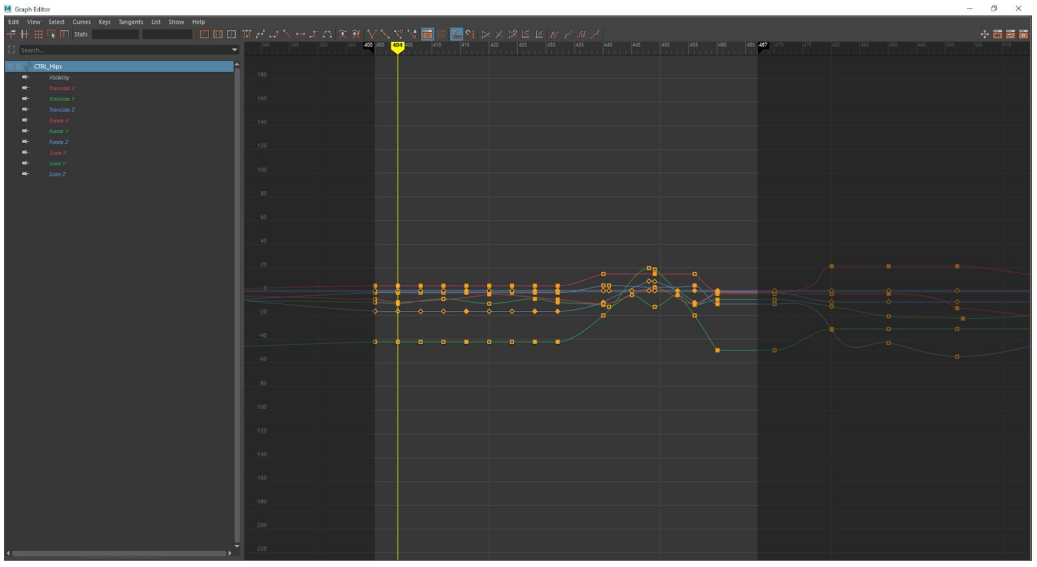


Fig. 60 Curvas de Animacion

Fase de Polish

En esta fase es donde pulimos la animación utilizando las curvas de animación que nos proporciona Maya, con esto podremos enfatizar gestos, acelerar objetos o añadir exageración para darle un carácter único.

QTE vs Cycle

Las animaciones para videojuegos acostumbran a comenzar y terminar en una misma postura, es lo que se conoce como un bucle o ciclo de animación, tal como se hace con el Walk Cycle, nuestro personaje comienza y acaba la acción en el mismo fotograma.

En este proyecto, al tratarse de una mezcla entre videojuego y cinemática, se ha utilizado tanto ciclos como animaciones tradicionales.

En el listado de animaciones están detalladas las animaciones que son tradicionales, con el nombre de QTE.

Para animar los QTE, debemos tener claro desde un principio donde estará la bifurcación entre acertar el QTE o fallarlo.

Para esto se elige un fotograma exacto en donde dividir los Animated Clips. desde donde saldrán o bien la animación de victoria o de derrota.

Exportación de Animaciones

Una vez acabadas las animaciones es hora de llevarlas al motor de videojuegos Unity. Para esto, seleccionamos nuestro **Join** Root, es decir el principal, y la Mesh a la cual está enlazado. Seleccionamos Exportar selección, seleccionamos el formato de archivo **FBX** y hacemos Bake de la animación seleccionando el rango en fotogramas que queremos exportar.

El Bake de animación "cocina" el movimiento que tenemos en los controladores directamente en los Joints, consiguiendo así que toda la animación se exporta tal y como la vemos incluyendo los cambios que hayamos realizado dentro del Graph Editor.

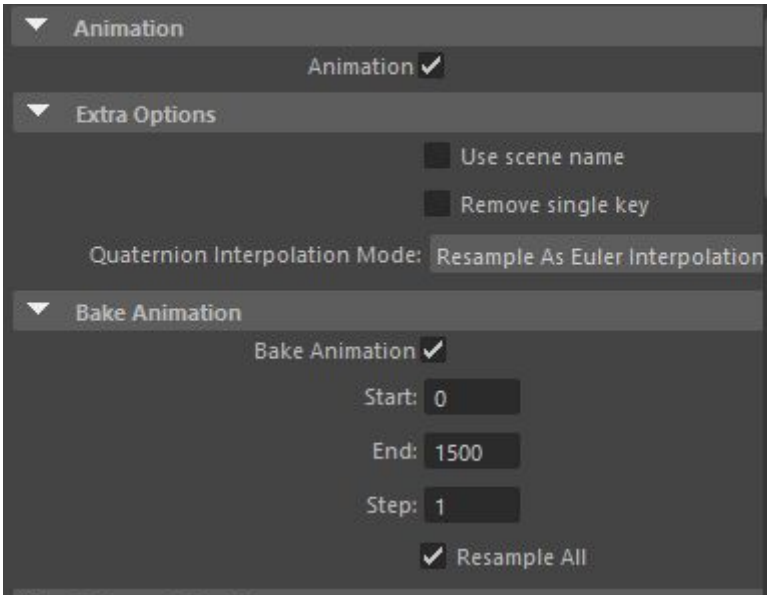


Fig. 61 Opciones de Bake

Unity

Una vez se han producido las animaciones es momento de importarlas en nuestro motor de videojuegos, en este caso **Unity**, y verificar si se reproducen correctamente, si no es así, se deberá volver a importar buscando el fallo y remediarlo.

Una vez tenemos las animaciones en nuestro motor, necesitamos crear sus clips de animación, para esto debemos abrir el **prefab** que **Unity** crea al importar un **FBX** y desde la pestaña de Animación crear los diferentes **Animation Clips** que tiene nuestra animación.
Para esto se selecciona un fotograma de inicio y un fotograma final, esto creará un **Animation clip** entre ese rango.

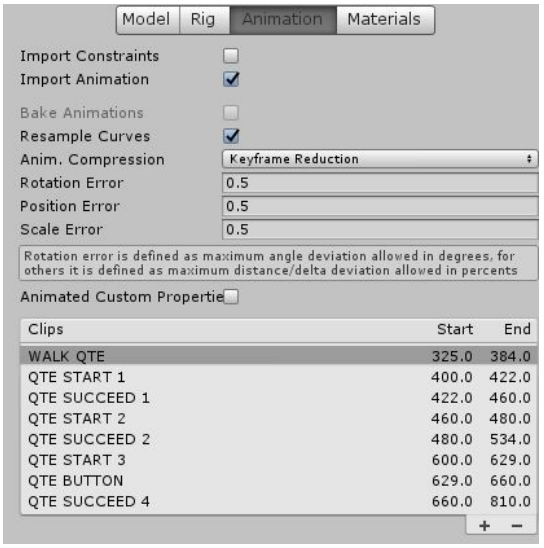


Fig. 62 Animation Clip

Animator Controller

El **Animator Controller** es un **Asset** de **Unity** que funciona como un director de animaciones, este es el encargado de unir las animaciones de una forma correcta y está representado como un conjunto de nodos que contienen animaciones, pudiendo hacer conexiones entre estas y controlar el flujo a través de condiciones que el usuario decida, en este caso tenemos las animaciones de "idle" y las de "run and Walk" además de un QTE que se utiliza para la animación de la puerta.

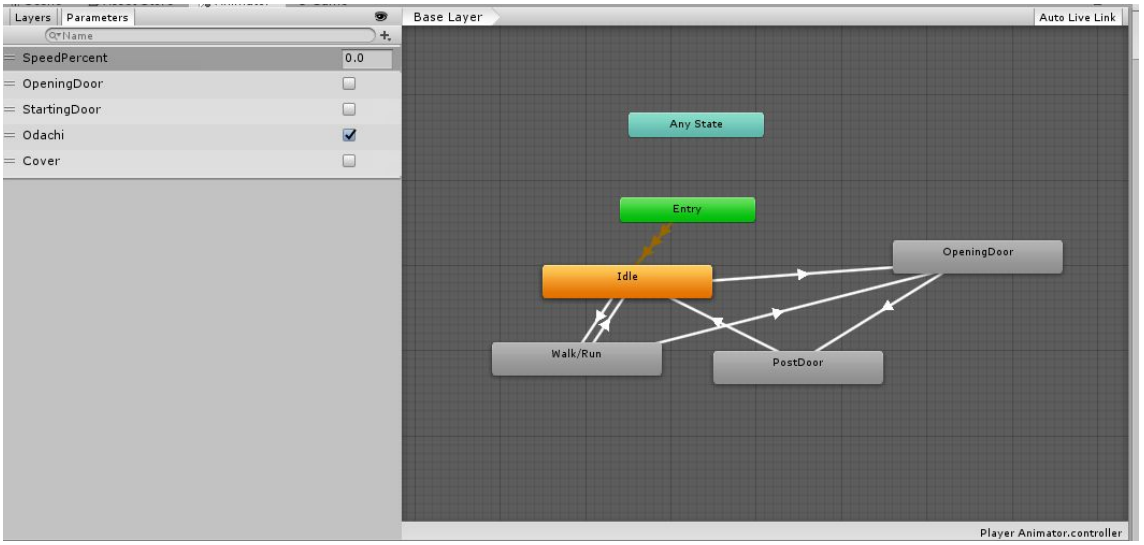


Fig. 63 Animator Controller

El animator controller se utiliza para las animaciones de ciclo, aunque tambien se podria utilizar para las animaciones de QTE, en este proyecto se utiliza la herramienta de unity Timeline, que es especial para realizar Cinemáticas, pero esto lo veremos más adelante.

Para que nuestro personaje se moviera además del Animator Controller también hemos tenido que crear un Script para poder desplazar a nuestro personaje ayudándonos de un componente llamado "Character Controller" que nos ofrece Unity.

Con esto conseguimos animar nuestro personaje guiándonos por la velocidad a la cual se mueve. Si la velocidad es 0, el personaje reproducirá la animación de "Idle" mientras que si se mueve a una velocidad de 2.5 unidades, este reproducirá la animación de "Walk". Si el personaje aumenta la velocidad hasta 6 unidades, este reproducirá la animación de "Run".

Todo esto se controla gracias al Animator Controller.

Además de esto también necesitamos una cámara para controlar la dirección de nuestro personaje utilizando un script.

Timeline

Como se ha comentado anteriormente, los QTE están programados a través de una herramienta de Unity llamada TImeline. Esta herramienta es similar a un programa de edicion de video, en la cual podemos utilizar una línea de tiempo para añadir animaciones, partículas, efectos de sonidos, scripts o activar y desactivar Assets.

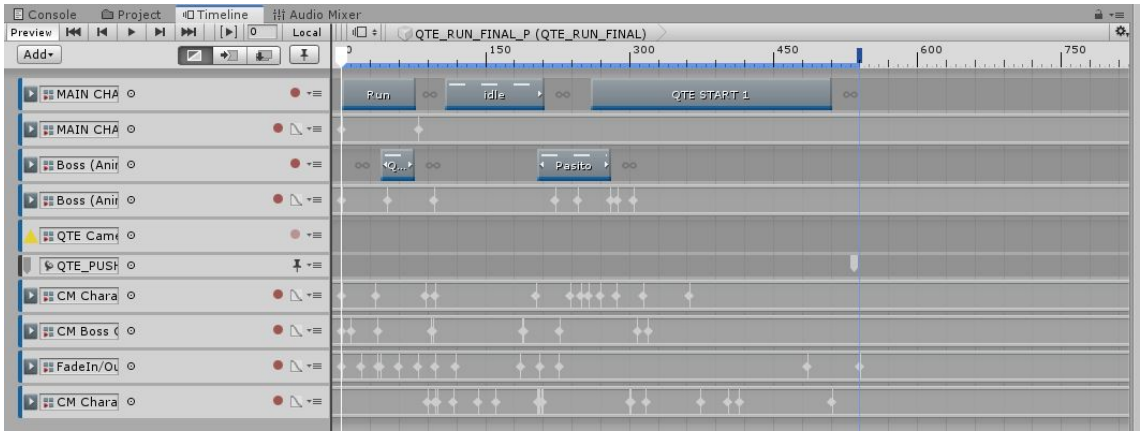


Fig. 64 Timeline

Utilizando timeline, podemos hacer un guión de las animaciones, eligiendo cuánto tiempo se reproducen, en que orden, la posición del personaje, etc.

Esto se lleva a cabo a través de las Animation Tracks que son las diferentes pistas de animación que se pueden ver en la imagen 64, en la cual cada Animation Track va asociada a un Gameobject que es el que realizará la animación.

Creando un QTE

Como hemos comentado antes, a la hora de crear los QTE, necesitamos diferentes animaciones que irán sincronizadas entre sí a través del Timeline de Unity.

En este proyecto hay un total de 6 QTE repartidos 3 en la zona final y 3 en el resto del mapa.

Los 3 Primeros QTE son independientes unos de otros, mientras que los últimos 3 son una secuencia.

Primeros pasos en Timeline

Para crear un QTE desde el principio a fin debemos crear un componente Playable Director dentro de un Gameobject vacío, este será el encargado de dirigir y sincronizar las diferentes Animation Tracks que decidamos utilizar dentro de la cinemática.

Una vez tenemos esto, debemos elegir que gameobjects se necesitaran dentro de la cinemática y crear un **Animation Track** por cada uno de estos **Gameobjects**. En el caso de Personajes animables, tales como **Oishi** o **Kira**, además de un Animation Track para las animaciones, también necesitaremos otro para modificar sus atributos, tales como rotación, traslación, escala, activar o desactivar colliders, etc.

Una vez hemos creado las Animation tracks, procedemos a seleccionar qué animated clips queremos que se reproduzcan.

Cuando hayamos seleccionado un Clip podemos darle a reproducir y veremos como nuestro personaje realiza la animación.

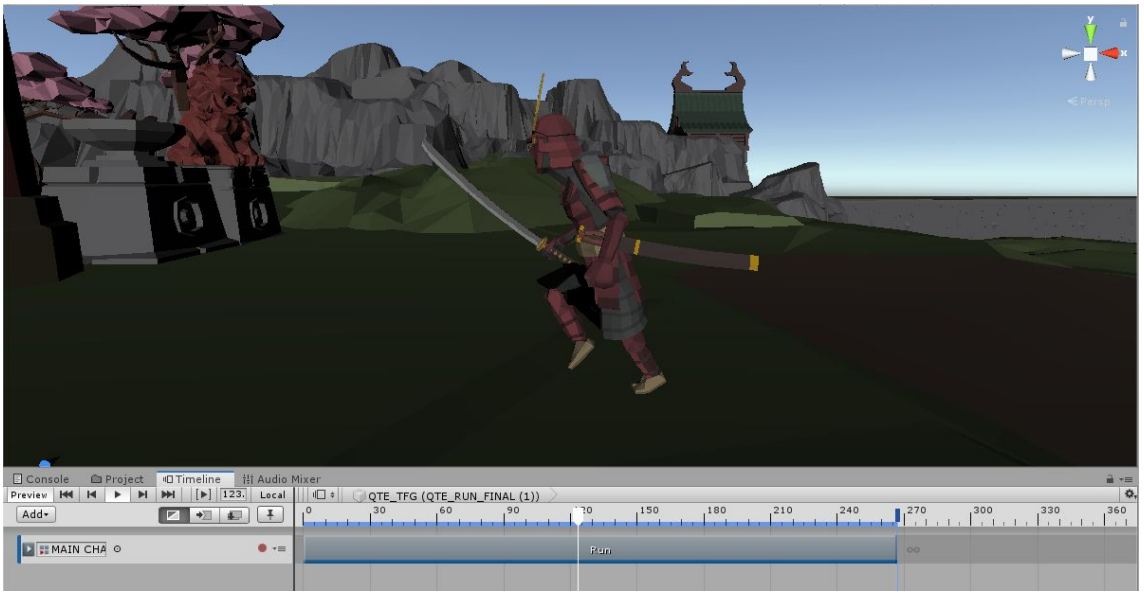


Fig.65 Fotograma de Animación Timeline

Ahora si queremos que nuestro personaje no solo reproduzca la animación sino que también se mueva, debemos crear otra Animation Track y volver a selecciona el personaje como GameObject que recibe la animación.

Para grabar un clip de movimiento tenemos que darle al botón de grabar (Círculo Rojo) y mover nuestro personaje a través del mundo.

Cuando lo tengamos en la posición inicial, debemos adelantar la línea de tiempo hasta el fotograma que decidamos y mover a nuestro personaje hasta la posición deseada, al acabar volvemos a pulsar el botón y Unity se encargará de realizar una interpolación entre estos dos puntos a la vez que reproduce la animación.

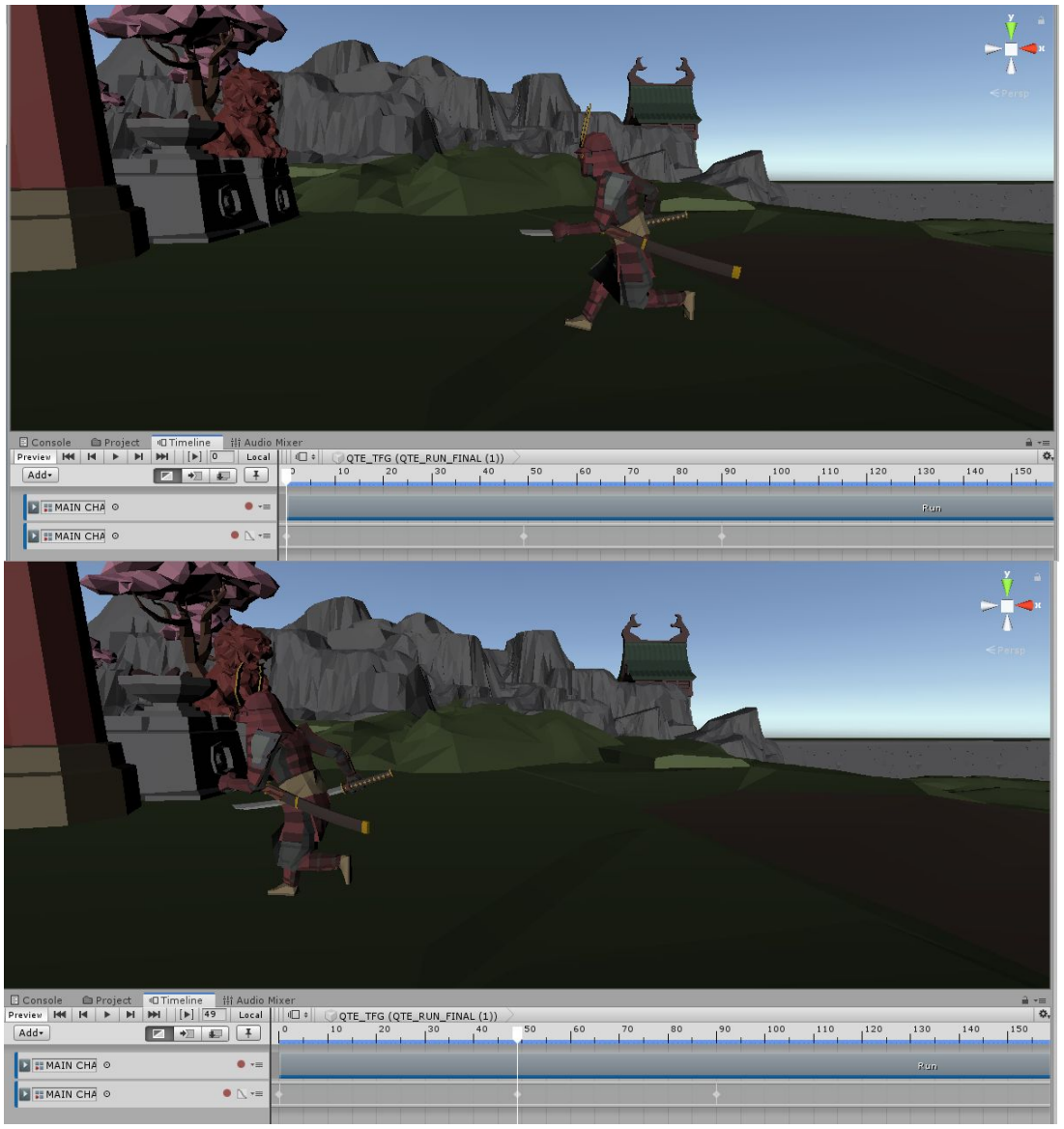


Fig. 66 Fotograma de Desplazamiento Timeline

Con esto tenemos un movimiento básico con una interpolación clásica, pero podemos retocar las curvas de interpolación haciendo click encima del botón de curvas que se encuentra justo al lado del de grabación.

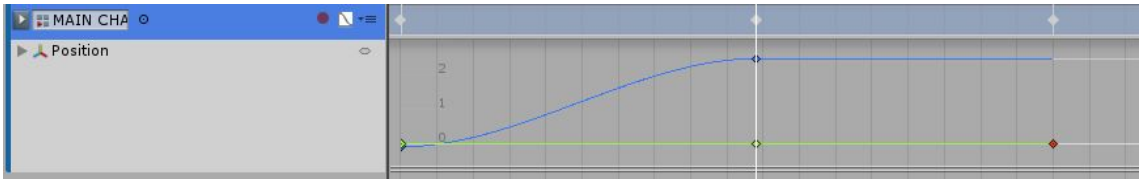


Fig. 67 Curvas de Animacion Timeline

Ahora ya tenemos todo lo necesario para realizar cinemáticas, pero un QTE es una cinemática interactiva, por lo que necesitamos poder modificar el resultado de la cinemática a través del input del mando.

Para ello primero vamos a ver cómo configurar un mando de Xbox en este caso para que responda conforme nosotros queremos.

Configuración del mando

Para poder configurar un mando de Xbox dentro de Unity debemos acceder a las opciones de Input dentro del motor. Para esto seleccionamos la ventana Edit->Project Settings. Aquí podemos realizar todos los ajustes del proyecto de Unity que tenemos, pero ahora nos centraremos en el Input.

Aquí veremos diferentes tipos de Input y debemos elegir la cantidad de botones que necesitamos, en este caso se ha utilizado un total de 18.

Para poder acceder al input del mando debemos conocer el **mapeado** que este presenta, cada botón está representado con un número que vemos en la siguiente imagen.

Fig. 68 Mapeado del mando de Xbox

Dentro de la ventana de un botón podemos modificar su nombre, el boton del mando a cual pertenece, al sensibilidad y el tipo de eje, entre otras cosas, a continuación hay un ejemplo de la configuracion del boton A y de uno de los Joysticks.

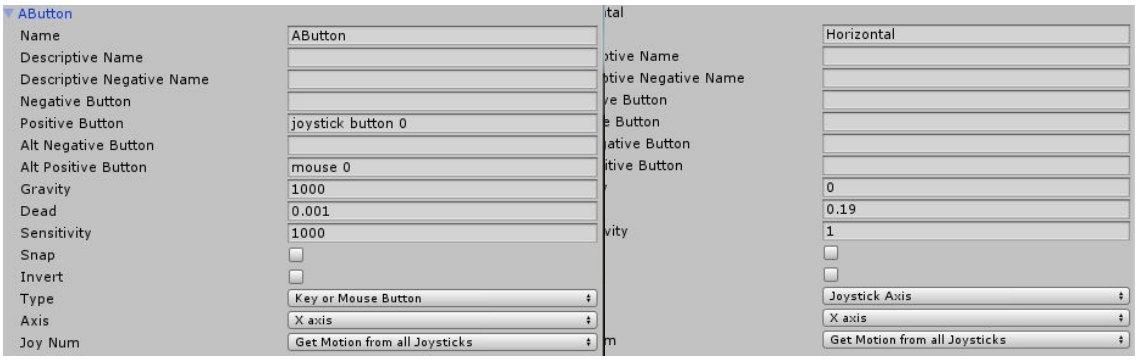


Fig. 69 Opciones de Input

Una vez hemos acabado de configurar todos los botones es hora de preparar un script para detectar el input y realizar una acción a través de Timeline.

QTE Manager

El QTE Manager es un Script que se ha creado para dirigir el orden en el cual se reproducen las cinemáticas. Este Script básicamente recibe una lista de Playables Director y dependiendo del input que recibe(acertar o fallar el QTE) reproduce o bien el QTE de Victoria o bien el de Derrota. El botón que se ha de pulsar es aleatorio entre los 4 botones principales del mando(A.B.X.Y).

```
public class QTE_Repeat : MonoBehaviour
{
    public float TimeThresh = 0;
    public GameObject button;
    public bool QTE = false;
    public int index;
    public string ButtonName;
    public PlayableDirector CurrentAnimation;
    public PlayableDirector NextAnimation;
    public List<PlayableDirector> playablesDirectors;
    public List<GameObject> QTE_Buttons;
    // Update is called once per frame
    void Update()
    {

        if (QTE == true)
        {
            button.SetActive(true);
            gameObject.SetActive(true);
            if (Input.GetButtonDown(ButtonName))
            {
                OnSuccess();
            }
        }
    }

    public void OnSuccess()
    {
        CurrentAnimation.Stop();
        NextAnimation.Play();
        CurrentAnimation = NextAnimation;
        Debug.Log("SUCCESS");
        button.SetActive(false);
        QTE = false;
    }
    public void Onfail(int AnimationFail)
    {
        NextAnimation = playablesDirectors[AnimationFail];
        CurrentAnimation.Stop();
        NextAnimation.Play();
        CurrentAnimation = NextAnimation;
        Debug.Log("Fail");
        button.SetActive(false);
        QTE = false;

    }
    public void StopAnimation()
    {
        foreach (PlayableDirector playableDirector in playablesDirectors)
        {
            playableDirector.Stop();
            QTE = false;
        }
    }

    public void StartAnimation()
    {
        CurrentAnimation.Play();
    }
    public void StartQTE(int i)
    {

        QTE = true;
        index = Random.Range(0, 3);
        NextAnimation = playablesDirectors[i];

        switch (index)
        {
            case 0:
                button = QTE_Buttons[0];
                ButtonName = "AButton";
                break;

            case 1:
                button = QTE_Buttons[1];
                ButtonName = "BButton";
                break;

            case 2:
                button = QTE_Buttons[2];
                ButtonName = "XButton";
                break;

            case 3:
                button = QTE_Buttons[3];
                ButtonName = "YButton";
                break;

            default:
                button = QTE_Buttons[0];
                ButtonName = "AButton";
                break;
        }
        button.SetActive(false);
    }
}
```

Fig. 70 Script de QTE Manager

Ahora bien, para acceder a estes script en el momento deseado debemos utilizar una función de Timeline que veremos a continuación.

Opciones avanzadas de Timeline

Ahora que tenemos las 3 cosas necesarias para realizar un QTE(Animaciones, Un script que las controle y un mando para introducir el Input) es hora de conectarlos entre sí.

Para esto se utiliza una herramienta que nos ofrece **Unity** solamente a partir de su versión 2019.1 llamada **Signal Emitter.**

Un **Signal Emitter** no es nada más que una señal que manda **Timeline** en el momento exacto que queremos y realiza una acción dentro de un **gameobject**, como si de un botón de unity se tratase que solo se pulsa en el momento exacto que el usuario elija.

Gracias a esto, podemos elegir el momento exacto en el cual nuestro QTE se activará, o bien, elegir en qué momento se acaba nuestro QTE. El Signal Emitter es como una Animation Track de Timeline pero que envía señales al Gameobject asociado, el tipo de señal que recibe se puede modificar como el usuario quiera además de modificar que es lo que hará el gameobject en el momento de recibir la señal a través de un **Signal Receiver**

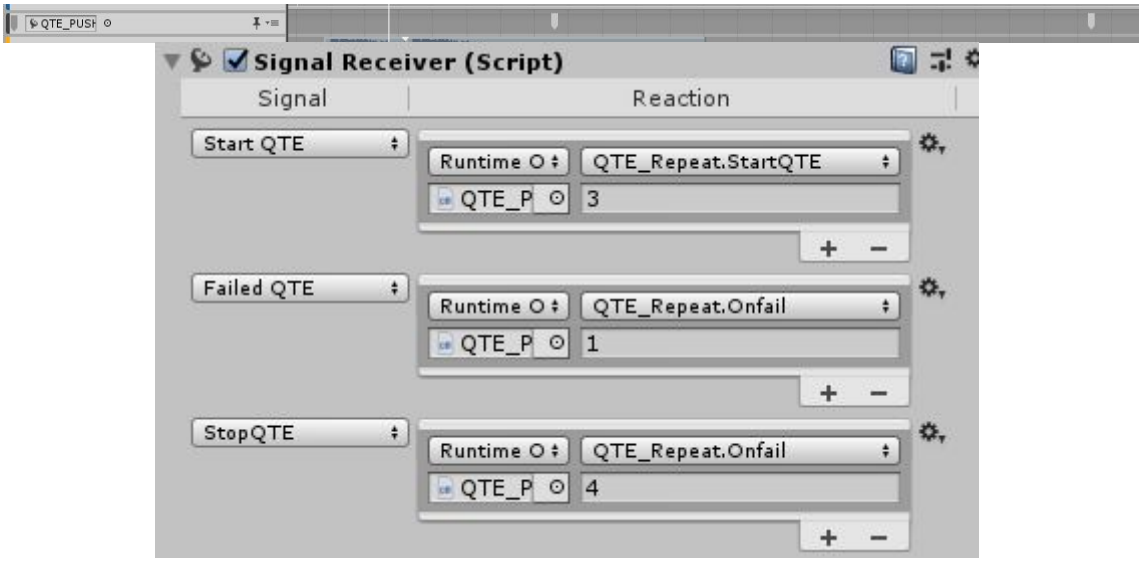


Fig. 71 Signal Emitter y Receiver

En el apartado de Resumen de QTE recapitulamos todo lo necesario para producir una cinemática de QTE. Ahora veremos el resto de funciones que nos ofrece Timeline.

Activation Track

Un Activation Track nos permite activar o desactivar gameobjects a través del timeline, además de seleccionar el estado en el cual seguirá el Gameobject después de reproducirse la pista.

Audio Track

Un Audio Track nos permite reproducir pistas de audio a traves del Timeline. Podemos controlar su volumen y en el caso de ser Stereo en que altavoz reproducirse.

Control Track

Un Control track nos permite controlar el tiempo de otros gameobjects, por ejemplo podríamos controlar el tiempo de Otro Playable Director a través de este o cambiar el tiempo de un Sistema de Partículas, además nos permite hacer instancias de Prefabs.

Playable Track

Un Playable Track nos permite modificar clases del tipo Playable.

Resumen de QTE

Para realizar un QTE se necesitan todo lo nombrado anteriormente, aunque en este apartado se realizará un breve resumen de los pasos a seguir.

Lo primero que necesitamos para un QTE son las animaciones de la cinemática, ya pueden ser tanto animaciones de personajes o simplemente de objetos como es el caso de la animación de "Abrir el Portón".

Una vez tenemos las animaciones, tenemos que crear sus Animation Clips, para separarlas en dos partes, la parte antes del QTE, y la parte despues del QTE.

Cuando tenemos las animaciones separadas, procedemos a crear un Playa para cada cinemática. Lo mas comun seria tener 3, una para antes del QTE, y dos para después, una para el acierto y otra para el fallo.

Cuando tenemos la cinemática en los Playable Director procedemos a crear un Script que gestione la reproducción de estas(QTE Manager).

Creamos el **Input** del mando, este paso se puede obviar si solo se jugara con teclado y ratón.

Dentro del Timeline creamos los Singal Emitters que le dirán a nuestro Script en qué momento comienza y en qué momento acaba nuestro QTE. Este evaluará el input que recibe y reproduce la cinemática correspondiente al resultado.

Con todos estos pasos habremos creado un QTE básico, ahora el usuario puede mezclar estos pasos de la manera que quiera para crear QTE más complejos como el combate final de este proyecto, el cual consta de un total de 7 Playables Directors y múltiples Tracks dentro de este.

Fig. 72 Playable Director

Ahora que tenemos el mapa creado, un personaje que puede navegar por este, los QTE creados, es hora de pasar a la fase de PostProducción, donde le daremos los últimos retoques y procederemos al cierre del Proyecto.

PostProduccion

En la fase de PostProducción lo primero que haremos será cerrar el ciclo del juego.

Para esto, solo tenemos que reproducir las cinemáticas en el orden que queramos.

Para esto simplemente tenemos que activar y desactivar las cinemáticas a través del Timeline antes mencionado o bien a través de un colisionador de Unity.

Música y Efectos de Sonido.

La Bandas sonora de Bushido ha sido compuesta y producida por **Cándido Ferrio Díaz.**

Para implementar la música dentro de Bushido se ha utilizado la herramienta de Unity Audio Source, en esta podemos añadir pistas de sonido de la manera que queramos.

La banda sonora está compuesta de 7 piezas que se unen en una sola pista de audio.

Los efectos de Sonidos son extraídos del banco de Imágenes y Sonidos del Estado Español.

Estas pistas de efectos de sonidos son implementadas dentro de los QTE a través de los Audio Tracks que se ha explicado anteriormente.

Fig. 73 Audio Track

Control de Cámaras

Si utilizamos la cámara que teníamos anteriormente, que es controlada con los joysticks del mando, la sensación de que es una cinemática no está presente.

Por lo cual se han creado una serie de Cámaras virtuales que van moviéndose y cambiando entre ellas para darle una sensación de cinemática real.

Para esto se ha utilizado la extensión de Unity Cinemachine.

Cinemachine

Cinemachine es una herramienta de Unity que nos permite tener más control sobre las cámaras dentro del videojuego.

Podemos crear cámaras que sigan a un gameobject, cámaras de control de personaje, cámaras estáticas o cámaras que siguen un trayecto.

Para nuestro proyecto se han utilizado cámaras virtuales que siguen a un gameobject, esta función se puede utilizar en el inspector de la cámara virtual.

Además las cámaras se han animado utilizando Timeline consiguiendo un resultado fluido tanto en el movimiento como en el cambio de cámaras.

Fig. 74 Virtual Camera

Partículas y UI

Para darle un toque más realista a los combates, se han creado partículas de chispas en los choques entre las katanas, para esto se ha utilizado el Particle System de Unity y se han activado y desactivado a través de TimeLine.

Para la **User Interface** se ha utilizado la opción genérica de UI que nos ofrece Unity, creando un Canvas y dentro de este imágenes de los botones de Xbox para que el jugador sepa qué botón ha de pulsar dentro del QTE.

Build

Para conseguir un ejecutable de nuestro juego, se ha utilizado la opción Build de Unity que nos permite elegir qué escenas queremos que se reproduzcan en el ejecutable que obtenemos al acabar.

El proyecto cuenta con un total de 4 Escenas que son el Menú, La introducción, el mapa final y por último la escena cinemática del combate final.

Al acabar tendremos una carpeta ejecutable que será nuestro proyecto acabado y listo para distribuir.

Playtesting

Al acabar nuestro proyecto se ha realizado playtesting tanto interno como externo del producto final, este se ha realizado con un total de 3 personas, siendo el resultado final de satisfactorio y sin errores importantes.

Se han detectado pequeños bugs visuales que se han intentado corregir sin éxito.

Contingencias

Para que este proyecto se pudiese completar se han tenido que aplicar unas cuantas contingencias, la más clara es la mezcla entre cinemática y videojuego que finalmente se ha optado por utilizar.

Además se ha realizado una adaptación del storyboard para disminuir la carga de trabajo, eliminando personajes y animaciones que no se podían conseguir. Pero la idea final de el asalto al castillo de Kira se ha mantenido intacta consiguiendo un resultado más que satisfactorio.

6.Conclusiones y trabajos futuros

Al comienzo de este trabajo no sabía cómo hacían las grandes empresas para producir una cinemática dentro de un motor de videojuegos que corre todo a tiempo real, ni mucho menos cómo conseguían hacer que se pudiera interactuar con ellas.

Analizando los objetivos iniciales de este proyecto, creo que se han conseguido cumplir todos los que no se han descartado por contingencias como perfeccionar las técnicas de modelado 3D. A pesar de esto ahora tengo una visión mucho más detallada de cómo se consiguen estas cinemáticas, ahora bien, me he dado cuenta de que comportan un trabajo de animación, Storytelling y de producción muchísimo más elevado del que creía y ahí es donde he tenido que aplicar unas cuantas contingencias para poder realizar este proyecto. A pesar de esto, he aprendido mucho sobre las cinemáticas y de su proceso de creación, he descubierto herramientas de Unity tales como TimeLine o Cinemachine que han sido vitales para este proyecto y he ido avanzando correctamente a través del desarrollo para finalmente tener una cinemática de QTE que podría tratarse perfectamente de un videojuego mucho más grande.

Creo que si volviese a realizar el proyecto con los conocimientos que tengo ahora mismo conseguiría un resultado de calidad aún mayor.

Por eso creo que una buena ampliación de este proyecto sería realizar un videojuego contando la historia completa de los 47 Ronin no solo el asalto al castillo, manteniendo la jugabilidad a base de QTE que creo que es una mecánica que se ha explorado poco para todas las opciones que presentan y solo unos pocos juegos han conseguido realmente realizar un buen trabajo con esta mecánica.

URL:

https://drive.google.com/file/d/1yAcl_ACiJZKQvzDx3chRFnawFcwM1r02/view

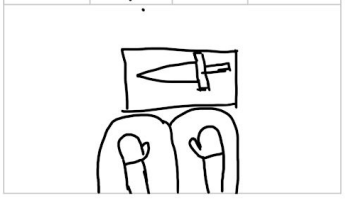
7. Bibliografia y

- 47 Ronin. *es.wikipedia.org/wiki/47_r%C5%8Dnin.* Consultado . 12,Feb, 2019
- Oishi Yoshido.en.wikipedia.org/wiki/%C5%8Cishi_Yoshio. Consultado 12,Feb,2019
- Physically based Rendering.en.wikipedia.org/wiki/Physically_based_rendering consultado 5,Mar,2019
- Zuzana Bubenová. 2016.Texturing a 3D Character in Hand-painted Style, Helsinki Metropolia University of Applied Sciences.
- Richard Williams. 2001. *The animator's Survival Kit.*ISBN 0-571-23833-0
- Jonathan Cooper. 2019. Game Anim Video Game Animation Explained. ISBN 978-1-1-3809-487-1
- Art of Blocking in Your Map. *www.worldofleveldesign.com/categories/level_design_tutorials/art_of_blocking_in_your_map.php.* Consultado. 15,Ago,2019.
- Hugues Barlet , Block design in level design www.gamasutra.com/blogs/HuguesBarlet/20140907/225061/Block_design_in_level_design.php. Consultado. 15, Ago,2019.
- Dan Milligan, Uncharted 3 Storyboards. https://www.behance.net/gallery/11927283/Uncharted-3-Storyboards. Consultado. 25,Ago,2019.
- Unity, Timeline. https://docs.unity3d.com/Manual/TimelineSection.html?_ga=2.57755630.510522083.1567969617-1476613571.1560351663. Consultado 10,Ago,2019.
- Adam Bradford,Parenting And Constraining In Maya. http://www.tallestanimator.com/AdamBradford_ConstrainingForAnimators.pdf. Consultado 26,Ago,2019.
- KeyFrame MP, http://zurbrigg.com/keyframe-mp. Consultado 1,Sept,2019.

8.Anexos

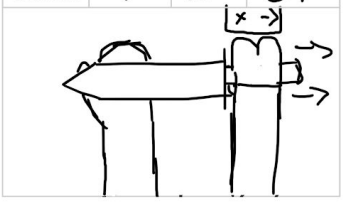
Storyboard

Título: Bushidō					Pág: 1
Escena No.	1	QTE	NO		



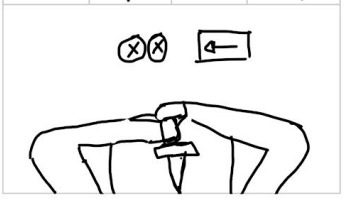
Empieza la cinematica con Asano preparado para realizar el Seppuku

Escena No.	1	QTE	SI



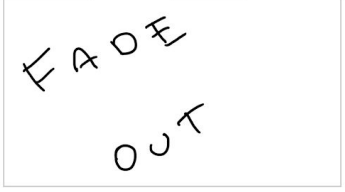
Asano desenvaina la daga

Escena No.	1	QTE	SI



Asano empuja la daga contra su estomago y lo desliza hacia la izquierda cometiendo Seppuku

Escena No.	1-2	QTE	NO



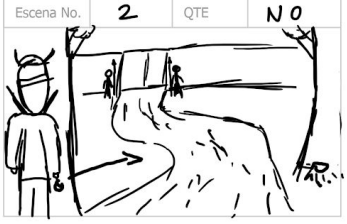
Fundido a negro y un texto nos cuenta la historia de los 47 Ronin

Escena No.		QTE	

Escena No.		QTE	

Fig. 52 Página 1 del Storyboard

Título: Bushidō					Pág:
Escena No.	2	QTE	NO		



Comienza la cinematica con Oishi escondido detras de un arbol, lanza una piedra hacia el otro extremo del camino y los guardias lo escuchan

Escena No.	2	QTE	NO

Oishi desenvaina la katana y se dirige por la izquierda hacia los guardias

Escena No.	2	QTE	SI

Al llegar al muro corre hacia el primer guardia.

Escena No.	2	QTE	SI

Ejecuta al primer guardia desde la espalda y se enfrenta al segundo

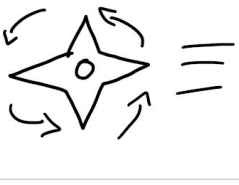
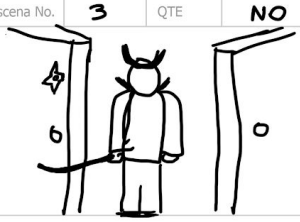
Escena No.	2	QTE	SI

Una vez ha acabado con los guardias y consigue la llave se dispone a abrir el portón.

Escena No.		QTE	

Fig. 53 Página 2 del Storyboard

Título: Bushidō	Pág:

Escena No. 3	QTE NO	Escena No. 3	QTE NO	Escena No. 3	QTE NO
					
La camara enfoca a un shuriken que viaja hacia Oishi		El shuriken falla y da en la puerta derecha del portón, Oishi mira al shuriken, desenvaina y mira hacia adelante		Oishi corre a traves del puente con una camara lateral	

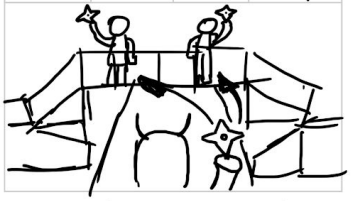
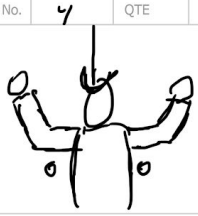
Escena No. 3	QTE SI	Escena No. 3	QTE SI	Escena No. 3	QTE SI
					
Oishi se enfrenta a 3 enemigos con la camara lateral		Una vez acaba con estos recoje dos shurikens del suelo y los lanza a dos enemigos que estaban en un puente.		Se dirije hacia la mansion y abre la puerta.	

Fig. 54 Página 3 del Storyboard

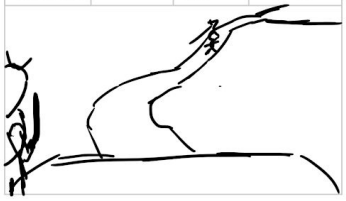
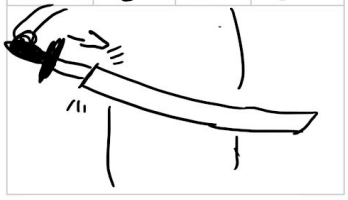
Título: Bushidō	Pág:

Escena No. 4	QTE NO	Escena No. 4	QTE SI	Escena No. 4	QTE SI
					
Al entrar en la mansion, Oishi ve a alguien subir por las escaleras, el cual cree que es Kira.		Sube las escaleras y llega a otra habitacion en la cual ha escuchado algo.		Dentro de la habitacion hay un armario en el cual se escuchan un llanto. Al abrirlo encuentra una dama que esta temblando y señalando a la ventana.	

Escena No. 4	QTE SI	Escena No.	QTE	Escena No.	QTE
					
Oishi ve a Kira saltando por la ventana y huyendo					

Fig. 78 Página 4 del Storyboard

Título: Bushidō	Pág: 5

Escena No. 5	QTE NO	Escena No. 5	QTE Si	Escena No. 5	QTE Si
					
Oishi salta por la ventana y ve a Kira subiendo por la montaña		Al llegar a lo alto de la montaña se encuentra a kira y combate contra el.		Al acabar con el le dedica la venganza a su amo y envaina su katana con una filigrana.	

Escena No. 5	QTE	Escena No.	QTE	Escena No.	QTE
					
Fundido a negro y salen un texto contandonos que pasa con oishi.					

Fig. 79 Página 5 del Storyboard

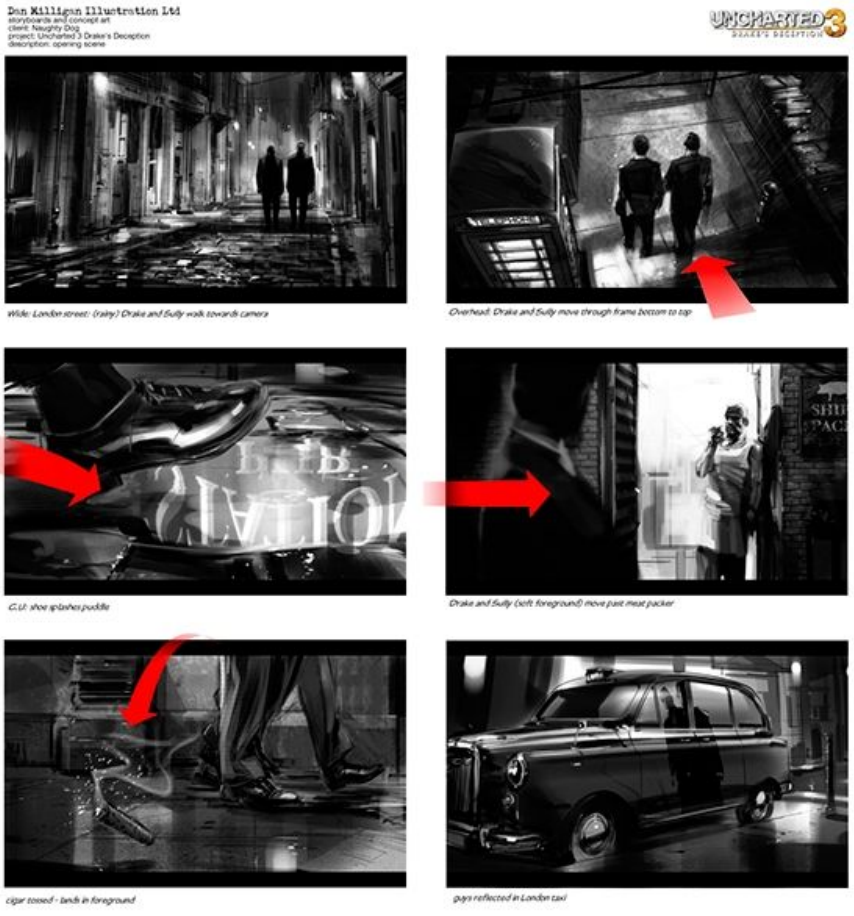


Fig. 80 Storyboard Uncharted 3